



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Gºen Ingeniería en Informática

Anexos:

**Plataforma Web para la realización de
torneos 2x2.**

Presentado por Jose Antonio Barbero Aparicio
en Enero de 2017

Tutor Jose Ignacio Santos Martín y Virginia Ahedo

Índice de contenido

Índice de ilustraciones.....	1
Índice de tablas.....	1
I -Plan de proyecto.....	2
1.Introducción.....	2
2.Planificación temporal.....	2
2.1.Sprint 1: 26/10/2016 – 2/11/2016....	3
2.2.Sprint 2: 3/11/2016 – 8/11/2016.....	5
2.3.Sprint 3: 8/11/2016 – 15/11/2016....	6
2.4.Sprint 4: 15/11/2016 – 22/11/2016...8	8
2.5.Sprint 5: 23/11/2016 – 29/11/2016.10	10
2.6.Sprint 6: 29/11/2016 – 6/12/2016..11	11
2.7.Sprint 7: 7/12/2016 – 14/12/2016. .13	13
2.8.Sprint 8: 14/12/2016 – 20/12/2016 14	14
2.9.Sprint 9: 21/12/2016 – 27/12/2016 15	15
2.10.Sprint 10: 28/12/2016 – 3/1/2017 16	16
2.11.Sprint 11.....	19
3.Estudio de viabilidad.....	20
3.1.Viabilidad legal.....	20
3.2.Viabilidad práctica.....	21
II -Especificación de requisitos.....	22
1.Introducción.....	22
2.Objetivos generales.....	22
3.Catálogo de requisitos.....	22
3.1.Requisitos funcionales.....	22
3.2.Requisitos no funcionales.....	23
4.Especificación de requisitos.....	24
4.1.Diagrama de casos de uso.....	24
4.2.Casos de uso.....	25
4.2.A.CU1 – Registro.....	25
4.2.B.CU2 – Log in.....	26
4.2.C.CU3 - Creación del torneo....26	26
4.2.D.CU4 - Finalización del torneo	27
4.2.E.CU5 - Participar en el torneo. 28	28
4.2.F.CU6 - Consultar resultados....29	29
III -Especificación de diseño.....	29
IV -Documentación técnica de programación	35
1.Introducción.....	35
2.Estructura de directorios.....	35
3.Manual del programador.....	36
3.1.A.NetLogo.....	36
3.1.B.XAMPP.....	38
4.Compilación, instalación y ejecución del proyecto.....	39
5.Pruebas del sistema.....	40
V -Documentación de usuario.....	44
1.Introducción.....	44
2.Requisitos de usuarios.....	44
2.1.Requisitos software.....	44
2.2.Requisitos hardware.....	44
3.Instalación.....	45
3.1.XAMPP.....	45
3.2.Java.....	45
3.3.Navegador.....	45
4.Manual de usuario.....	47
4.1.Inicio.....	48
4.2.Log in.....	48
4.3.Registro.....	49
4.4.Torneos.....	49
4.5.Crear torneo.....	50
4.6.Torneos finalizados.....	51
4.7.Participar.....	51
4.8.Finalizar torneo.....	53
	54
VI -referencias.....	55
Bibliografía.....	55

Índice de ilustraciones

Illustration 1: Velocity Tracking a día 28/12/2016.....	3
Illustration 2: Ejemplo de la gestión de un sprint con ZenHub.....	4
Illustration 3: burndown chart del sprint 1.....	4
Ilustración 4: Listado de tareas del sprint 1....5	5
Illustration 5: burndown chart del sprint 2.....6	6
Ilustración 6: Listado de tareas del sprint 2....7	7
Illustration 7: burndown chart del sprint 3.....7	7
Ilustración 8: Listado de tareas del sprint 3....8	8
Illustration 9: burndown chart del sprint 4.....8	8
Ilustración 10: Listado de tareas del sprint 4..9	9
Illustration 11: burndown chart del sprint 5. 10	10

Ilustración 12: Listado de tareas del sprint 5 11	11
Illustration 13: burndown chart del sprint 6. 12	12
Ilustración 14: Listado de tareas del sprint 6 13	13
Illustration 15: burndown chart del sprint 7. 14	14
Ilustración 16: Listado de tareas del sprint 7 14	14
Illustration 17: burndown chart del sprint 8. 15	15
Ilustración 18: Listado de tareas del sprint 8 15	15
Illustration 19: burndown chart del sprint 9. 16	16
Ilustración 20: Listado de tareas del sprint 9 17	17
Illustration 21: burndown chart del sprint 10	18
Illustration 22: Listado de tareas del sprint 10	19

Illustration 23: burndown chart del sprint 11	20
Illustration 24: Listado de tareas del sprint 11	21
Illustration 25: Diagrama de casos de uso....	25
Illustration 26: Tabla “users” de base de datos	31
Illustration 27: Tabla “participantes” de base de datos.....	32
Illustration 28: Tabla "torneos" de base de datos.....	32
Illustration 29: Tabla "salas" de base de datos	32
Illustration 30: Tabla "payoffs" de base de datos.....	33
Illustration 31: Diagrama de flujo de la vida de un torneo.....	34
Illustration 32: Modelo-vista-controlador[1]35	
Illustration 33: Nuevo experimento en NetLogo Behaviour Space.....	38
Illustration 34: Formulario de un nuevo experimento en NetLogo Behaviour Space.....	39
Illustration 35: Experimento creado en NetLogo Behaviour Space.....	40
Illustration 36: Servidores iniciados en XAMPP.....	41
Illustration 37: Ejecución de las pruebas.....	42
Illustration 38: Importar máquina virtual.....	49
Illustration 39: Página de inicio.....	50
Illustration 40: Log in.....	51
Illustration 41: Registro.....	51
Illustration 42: Torneos desde la perspectiva de alguien sin registrar.....	52
Illustration 43: Torneos desde la perspectiva de un usuario profesor.....	52
Illustration 44: Participación en el torneo....	53
Illustration 45: Código de la estrategia de un alumno.....	54
Illustration 46: Descarga del modelo.....	55
Illustration 47: Finalización de un torneo.....	55

Índice de tablas

Table 1: Caso de uso 1: Registro.....	25
Table 2: Caso de uso 2: Login.....	26
Table 3: Caso de uso 3: Creación de un torneo	26
Table 4: Caso de uso 4: Finalizar torneo.....	27
Table 5: Caso de uso 5: Participar en torneo	28
Table 6: Caso de uso 6: Consultar resultados	29
Table 7: Casos de prueba en el login.....	42
Table 8: Casos de prueba en el registro.....	42
Table 9: Casos de prueba en la inscripción a los torneos.....	43
Table 10: Casos de prueba en la finalización de torneos.....	43
Table 11: Casos de prueba en la creación de torneos.....	43
Table 12: Casos de prueba en la fase de test de la estrategia.....	44

I - PLAN DE PROYECTO

1. Introducción

En este apartado vamos a analizar el proceso de planificación de las tareas a lo largo del proyecto. Como se ha especificado en el apartado de la memoria correspondiente, para la gestión de todo el proceso se ha utilizado la extensión de GitHub ZenHub.

A continuación analizaremos la viabilidad del proyecto desde una perspectiva docente, al no tratarse de una aplicación comercial.

Si se desea, se puede consultar el proyecto en GitHub a través del enlace <https://github.com/jba0040/PlataformaTorneos2x2>. También se ha añadido al usuario ubutfgm al repositorio para facilitar el acceso al tribunal. Se recomienda instalar la extensión ZenHub si se desea acceder a funcionalidades como el tablero de tareas o los gráficos burndown.

2. Planificación temporal

A lo largo del proceso de desarrollo se ha seguido una metodología ágil de tipo Scrum. El periodo de desarrollo ha estado dividido en 11 sprints de una semana de duración cada uno. A continuación analizaremos estos sprints y las tareas que contenían, así como sus gráficas burn-down.

Respecto al tiempo asignado a cada tarea, ZenHub funciona a través de story points. Estos puntos se asignan a cada tarea estimando una duración dentro de los miembros de la sucesión de Fibonacci.

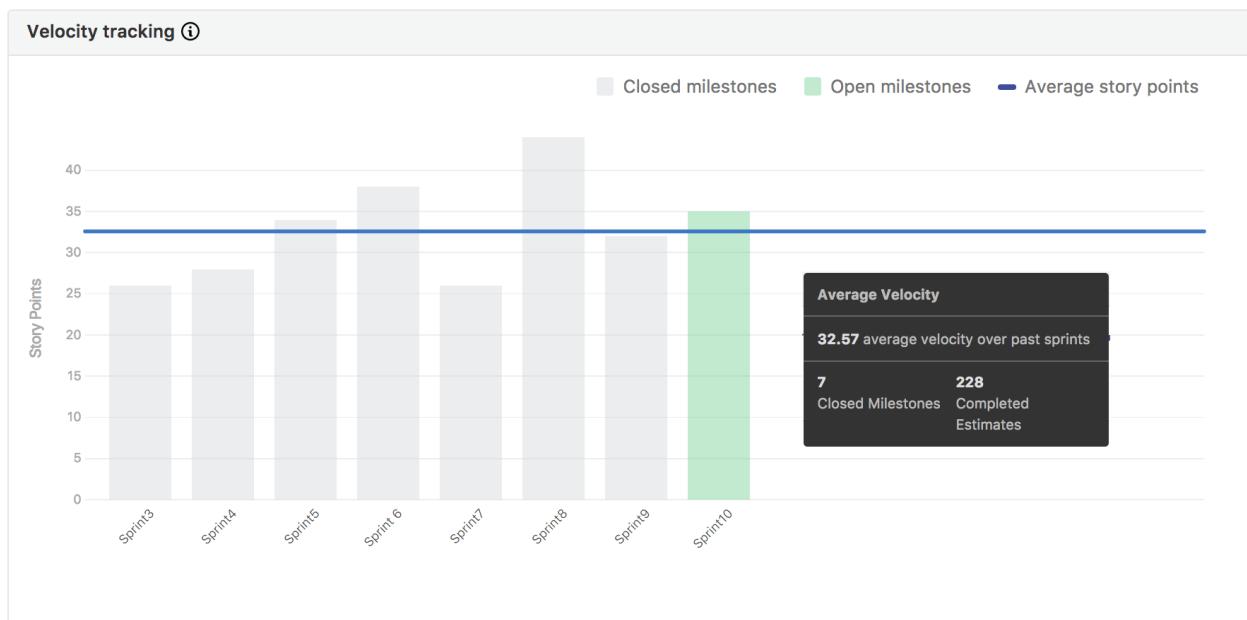


Illustration 1: Velocity Tracking a día 28/12/2016

Inicialmente podemos observar un gráfico de actividad que analiza los últimos sprints lla-

mado velocity tracking. Zenhub analiza a través de este gráfico los siete últimos sprints que hayan sido cerrados y obtiene una media de la velocidad de desarrollo por sprint. En el ejemplo que vemos, a día 18 de diciembre, la velocidad media para los últimos sprints es de 32,57 (línea azul). Vemos que, a excepción del sprint 7, las tareas van aumentando en tiempo poco a poco para luego, en los últimos sprints, descender ligeramente. Si pudiésemos observar este gráfico para todos los sprints veríamos un primer sprint con más horas que el resto, debido a la carga inicial que lleva el proyecto.

The screenshot shows the ZenHub interface with four boards:

- Backlog:** Contains tasks like "PlataformaTorneos2x2 #1 Diseño del Torneo" and "PlataformaTorneos2x2 #2 Introducción de estrategias de juego".
- Sprint planning:** Contains tasks like "PlataformaTorneos2x2 #81 Máquina virtual" and "PlataformaTorneos2x2 #70 Conclusiones y líneas de trabajo futuras".
- To do:** Contains tasks like "PlataformaTorneos2x2 #77 Anexos: Especificación de requisitos" and "PlataformaTorneos2x2 #78 Anexos: Especificación de diseño".
- In Progress:** Contains tasks like "PlataformaTorneos2x2 #76 Anexos: Plan de proyecto" and "PlataformaTorneos2x2 #79 Anexos: Documentación técnica de programación".

Illustration 2: Ejemplo de la gestión de un sprint con ZenHub

2.1. Sprint 1: 26/10/2016 – 2/11/2016



Illustration 3: burndown chart del sprint 1

El primer sprint es un periodo bastante distante del resto en cuanto a contenido debido a cierto desconocimiento de las herramientas, del tema a desarrollar, etc. Por ello este primer sprint está centrado en la investigación, el establecimiento de las bases iniciales, etc.

Las tareas que contiene están relacionadas con la creación del servidor, de la base de datos, de la página web básica... Pero la mayor parte del tiempo del sprint está centrada en la investigación acerca de la teoría de juegos y de las herramientas y procesos que se van a utilizar posteriormente en el proyecto.

Como vemos, en esta primera iteración, las tareas fueron terminadas casi de forma simultanea y al final del sprint. Esto denota un desarrollo paralelo de las actividades que se distancia de la línea más óptima de finalización de las mismas.

En la ilustración 4 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Página web base con login Código PlataformaTorneos2x2 #7	(8)
⌚ Creación del servidor Código PlataformaTorneos2x2 #8	(1)
⌚ Documentación del Sprint 1 Documentación PlataformaTorneos2x2 #9	(5)
⌚ Investigación sobre la teoría de juegos. Investigación PlataformaTorneos2x2 #10	(21)
⌚ Creación de la base de datos Código PlataformaTorneos2x2 #15	(3)

Ilustración 4: Listado de tareas del sprint 1

2.2. Sprint 2: 3/11/2016 – 8/11/2016



Illustration 5: burndown chart del sprint 2

Este sprint está centrado en el desarrollo del modelo inicial de NetLogo con el que posteriormente se espera que los alumnos prueben sus estrategias. Se desarrolló una versión inicial que enfrentase únicamente a dos estrategias y posteriormente se amplió para que se pudiese elegir una estrategia entre varias establecidas. Finalmente se fueron añadiendo estrategias para ampliar la variedad entre las preestablecidas.

Como vemos en el burndown chart, el progreso de finalización de las tareas es más progresivo que en el caso del sprint 1, acercándose más a la línea que marca un desarrollo óptimo.

En la ilustración 6 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Desarrollo del modelo primario en NetLogo Código PlataformaTorneos2x2 #16	(8)
⌚ Estrategia aleatoria Código PlataformaTorneos2x2 #17	(2)
⌚ Estrategia Tit for Tat Código PlataformaTorneos2x2 #18	(2)
⌚ Estrategia Always Defect. Código PlataformaTorneos2x2 #19	(1)
⌚ Estrategia Always Cooperate Código PlataformaTorneos2x2 #20	(1)
⌚ Estrategia Friedman Código PlataformaTorneos2x2 #21	(2)
⌚ Estrategia Tit for Two Tats Código PlataformaTorneos2x2 #23	(1)
⌚ Estrategia Joss Código PlataformaTorneos2x2 #24	(2)
⌚ Modelo inicial Código PlataformaTorneos2x2 #25	(5)
⌚ Documentación Sprint 2 Documentación PlataformaTorneos2x2 #26	(5)

Ilustración 6: Listado de tareas del sprint 2

2.3. Sprint 3: 8/11/2016 – 15/11/2016

**Illustration 7: burndown chart del sprint 3**

Esta iteración está orientada a la mejora del primer modelo de NetLogo con el objetivo de comenzar a profundizar en funcionalidades destinadas al modelo global.

En primer lugar, se optimizó el modelo que se había desarrollado en el sprint anterior para mejorar el rendimiento a través del uso de algunas funcionalidades de NetLogo que se habían pasado por alto.

En segundo lugar, se comenzó con la gestión de estrategias desde archivos permitiendo la lectura de las mismas, comenzando con estrategias prefijadas y continuando con la estrategia de libre definición del alumno.

En la ilustración 8 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Lectura de estrategias desde archivo Código PlataformaTorneos2x2 #27	(5)
⌚ Netlogo Behaviour Space Investigación PlataformaTorneos2x2 #30	(2)
⌚ Lectura de diferente número de modelos Código PlataformaTorneos2x2 #31	(8)
⌚ Optimización Código PlataformaTorneos2x2 #32	(8)
⌚ Lectura de la estrategia del alumno Código PlataformaTorneos2x2 #33	(3)

Ilustración 8: Listado de tareas del sprint 3

2.4. Sprint 4: 15/11/2016 – 22/11/2016

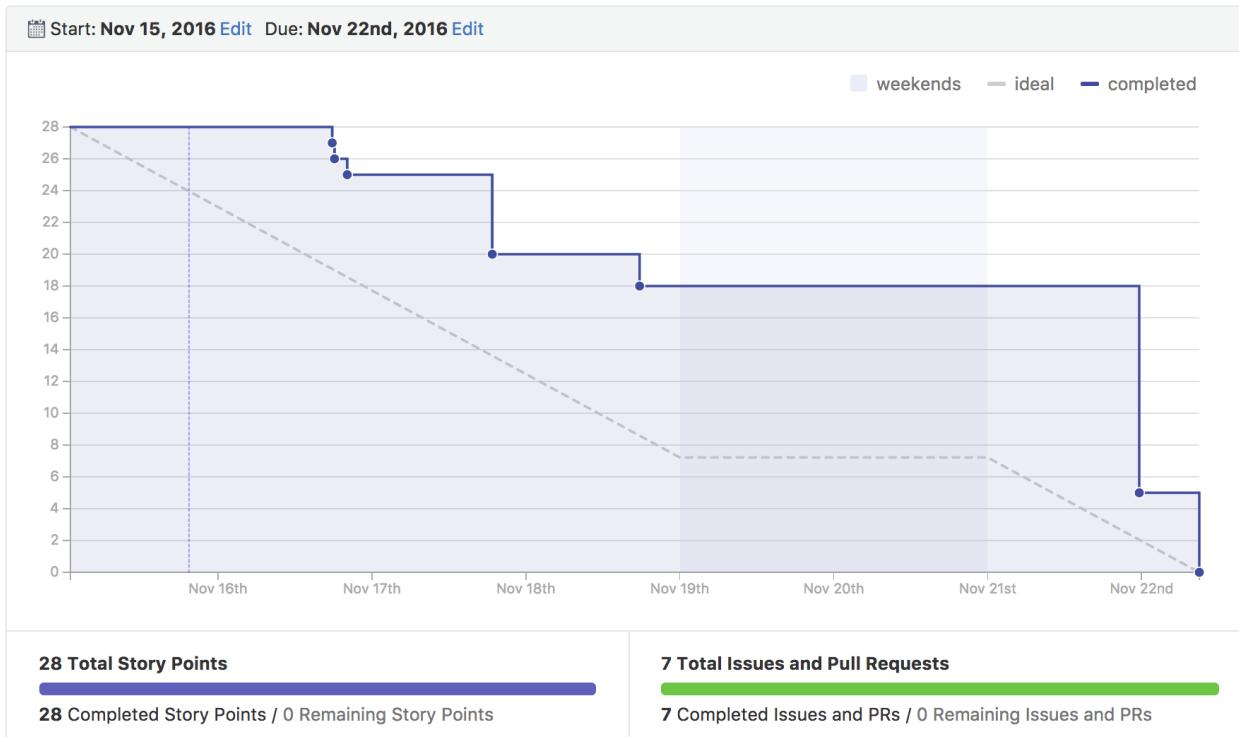


Illustration 9: burndown chart del sprint 4

Este sprint comienza a tratar tareas relacionadas con la web, habiendo realizado los modelos de NetLogo en la anterior iteración.

En cuanto al servidor, se configura la base de datos para los participantes, torneos, salas de

torneo, etc. Respecto a la página web, se desarrolla el acceso a los torneos y se adapta el modelo de prueba de estrategias a su versión web. En esta última tarea se encontraron algunas dificultades y surgieron varios bugs relacionados con el modelo web que fueron finalmente eliminados en ese mismo sprint.

En la ilustración 10 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Modelo Web Código PlataformaTorneos2x2 #34	(13)
⌚ Base de datos: Participantes Código PlataformaTorneos2x2 #35	(1)
⌚ Página de entrada al torneo Código PlataformaTorneos2x2 #36	(5)
⌚ Base de datos: Torneos Código PlataformaTorneos2x2 #37	(1)
⌚ Base de datos: Salas de torneo Código PlataformaTorneos2x2 #38	(1)
⌚ Adaptar modelo a web bug Código PlataformaTorneos2x2 #39	(2)
⌚ Adaptar modelo web a HTML5 bug Código PlataformaTorneos2x2 #41	(5)

Ilustración 10: Listado de tareas del sprint 4

2.5. Sprint 5: 23/11/2016 – 29/11/2016



Illustration 11: burndown chart del sprint 5

Las tareas de este sprint están, en su mayoría, destinadas a ampliar la funcionalidad de la web. Se profundiza más en la funcionalidad del modelo web, permitiendo almacenar las estrategias de los alumnos una vez las hayan probado y compilado. Respecto al torneo, se añade la funcionalidad que permite exportar los resultados en un fichero, para poder almacenarlas posteriormente.

Sin embargo, la tarea principal de este sprint es la funcionalidad asociada a la finalización del torneo. Se trata de una tarea complicada ya que involucra una gran cantidad de ficheros del servidor, incluyendo modelos “.nlogo”, estrategias “.nls”, archivos binarios, bibliotecas de NetLogo, etc. Además surgieron algunos problemas de permisos en los ficheros que ralentizaron el desarrollo.

En la ilustración 12 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Guardado del modelo del alumno en el servidor Código PlataformaTorneos2x2 #40	(8)
⌚ Sesiones Código PlataformaTorneos2x2 #46	(3)
⌚ Finalización de torneo. Código PlataformaTorneos2x2 #47	(13)
⌚ Guardado de modelo con el numero de participante Código PlataformaTorneos2x2 #48	(3)
⌚ Retorno de resultados en fichero. Código PlataformaTorneos2x2 #49	(5)
⌚ Eliminación de gráfica bug PlataformaTorneos2x2 #50	(1)
⌚ Redirección tras guardar archivo bug PlataformaTorneos2x2 #51	(1)

Ilustración 12: Listado de tareas del sprint 5

2.6. Sprint 6: 29/11/2016 – 6/12/2016



Illustration 13: burndown chart del sprint 6

En este sprint, en primer lugar, se continúa trabajando sobre la funcionalidad de la finalización del torneo. En este caso se añade un control de confirmación y de tipo de usuario para que solo los profesores puedan finalizar los torneos.

Además se desarrollan muchas otras tareas de menor duración entre las que encontramos la solución de un bug que impedía que el modelo de NetLogo en la web fuese responsive. Además se añade una lista de los torneos finalizados debajo de los activos para que los usuarios puedan consultarlos posteriormente. Respecto a los torneos, se añade la funcionalidad restante para la creación de los torneos por parte de los profesores, se hace que los resultados de los torneos ahora queden almacenados en base de datos y se muestran los participantes que hay en cada sala en cada momento.

Respecto a la web en sí misma, se crea un sistema de sesiones para profesores y usuarios y se hace que los torneos incluyan en sus datos una fecha de inicio y una fecha de finalización para poder ordenarlos posteriormente.

En la ilustración 14 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Responsive (Modelo NetLogo Web) bug Código PlataformaTorneos2x2 #45	(2)
⌚ Control de finalización de torneo Código PlataformaTorneos2x2 #52	(13)
⌚ Listado de torneos finalizados y activos. Código PlataformaTorneos2x2 #53	(3)
⌚ Almacenamiento de resultados en base de datos. Código PlataformaTorneos2x2 #55	(3)
⌚ Sistema de cierre de sesión Código PlataformaTorneos2x2 #56	(2)
⌚ Participantes actuales Código PlataformaTorneos2x2 #57	(3)
⌚ Control de creación de torneos Código PlataformaTorneos2x2 #58	(8)
⌚ Sistema de sesiones para los alumnos Código PlataformaTorneos2x2 #59	(1)
⌚ Permisos de finalización de torneos. Código PlataformaTorneos2x2 #61	(2)
⌚ Torneos: Fechas Código PlataformaTorneos2x2 #62	(1)

Ilustración 14: Listado de tareas del sprint 6

2.7. Sprint 7: 7/12/2016 – 14/12/2016



Illustration 15: burndown chart del sprint 7

Este sprint se caracteriza por tareas centradas en la finalización de las funcionalidades importantes del la plataforma web, y el comienzo del desarrollo de la documentación.

Las funcionalidades web desarrolladas en esta iteración son, en primer lugar, un sistema para consultar los torneos finalizados en general, o los de un usuario en particular y, en segundo lugar, un sistema de participación orientado al nuevo sistema de sesiones de los alumnos. Ahora es posible que un alumno participe tanto si está registrado como si no, la diferencia será en funcionalidad posterior para la consulta de los resultados de sus torneos. En ambos casos se deberá identificar, pero en el caso de estar registrado, la mayoría de los datos se llenarán automáticamente.

En cuanto a la documentación, se comienza a desarrollar por el apartado de técnicas y herramientas por ser el más accesible al principio para tomar contacto con la documentación y el sistema de referencias de Mendeley.

En la ilustración 16 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Consulta de torneos Código PlataformaTorneos2x2 #54	(3)
⌚ Participación de alumnos Código PlataformaTorneos2x2 #63	(2)
⌚ Resultados de torneos finalizados. Código PlataformaTorneos2x2 #64	(8)
⌚ Técnicas y herramientas. Documentación PlataformaTorneos2x2 #65	(13)

Ilustración 16: Listado de tareas del sprint 7

2.8. Sprint 8: 14/12/2016 – 20/12/2016

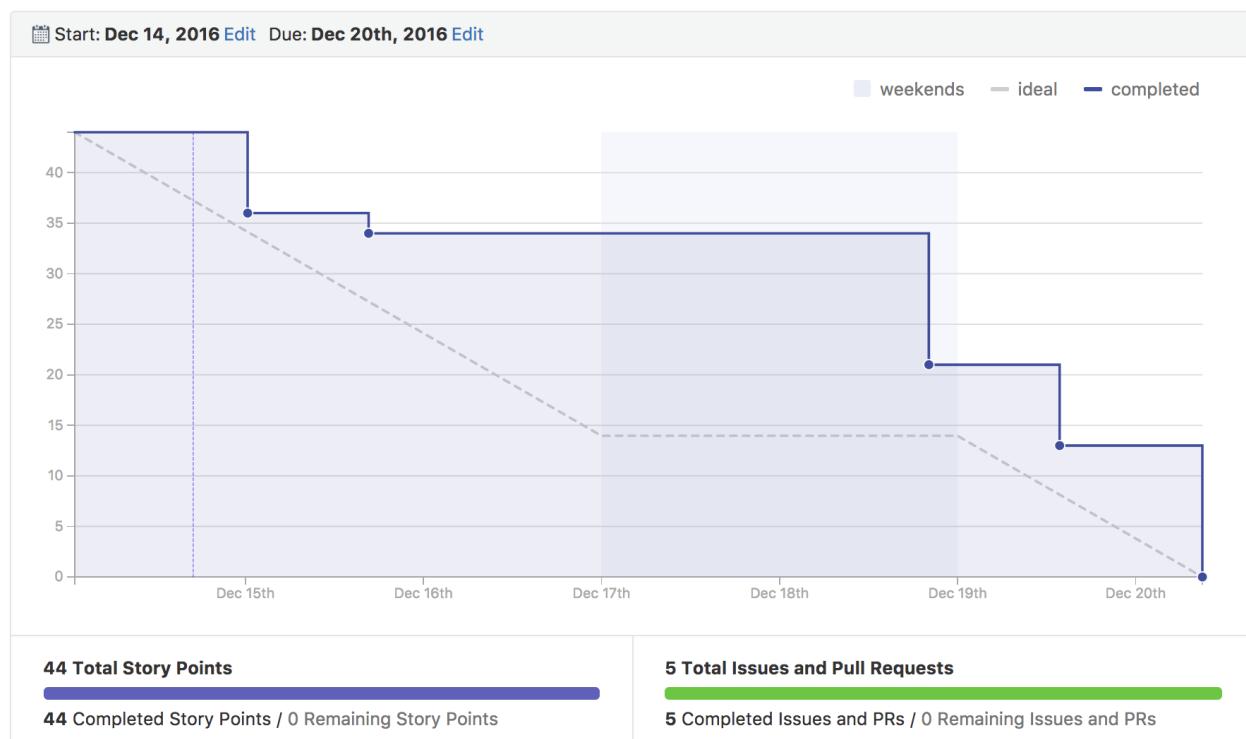


Illustration 17: burndown chart del sprint 8

En este sprint se profundiza definitivamente en el apartado de documentación, conllevando también cierto trabajo de investigación.

Se redacta el resumen inicial del proyecto, dejando la introducción para más adelante. También se realiza el apartado de objetivos y las referencias bibliográficas del punto sobre técnicas y herramientas, que había quedado pendiente del sprint anterior. Las partes con más esfuerzo requerido son los conceptos teóricos y los trabajos relacionados. Esto se debe a que se debe realizar un proceso de investigación sobre ambas.

En la ilustración 18 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Descripción del proyecto. Documentación PlataformaTorneos2x2 #28	(8)
⌚ Objetivos del proyecto. Documentación PlataformaTorneos2x2 #66	(8)
⌚ Conceptos teóricos. Documentación Investigación PlataformaTorneos2x2 #67	(13)
⌚ Trabajos relacionados. Documentación Investigación PlataformaTorneos2x2 #69	(13)
⌚ Referenciación de las técnicas y herramientas Documentación PlataformaTorneos2x2 #71	(2)

Ilustración 18: Listado de tareas del sprint 8

2.9. Sprint 9: 21/12/2016 – 27/12/2016

**Illustration 19: burndown chart del sprint 9**

A diferencia del anterior sprint, centrado únicamente en la documentación, en este se realiza parte de documentación y parte de desarrollo.

En cuanto a la documentación se llevan a cabo los apartados de aspectos relevantes del proyecto y la introducción.

Respecto al desarrollo, se añaden algunas funcionalidades que durante el proceso se dejaron para más adelante, con el fin de buscar la funcionalidad principal cuanto antes. Entre estos puntos encontramos el sistema de cifrado de claves para el registro y el login, y el sistema que permite fijar al profesor los parámetros de payoff en el momento de creación del torneo.

Como aspecto relevante, encontramos una tarea destinada a la resolución de un bug. Y es que, dado que se tienen que guardar las estrategias en archivos para que NetLogo las lea, se estaban guardando las de todas las salas en el mismo lugar, lo cual provocaba que unas sobrescribiesen a

las otras.

En la ilustración 20 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Aspectos relevantes del desarrollo del proyecto. Documentación PlataformaTorneos2x2 #68	(8)
⌚ Cifrado de claves Código PlataformaTorneos2x2 #72	(5)
⌚ Permitir al profesor fijar los payoffs. Código PlataformaTorneos2x2 #73	(8)
⌚ Introducción Documentación PlataformaTorneos2x2 #74	(3)
⌚ Diferenciación del estrategias por salas bug Código PlataformaTorneos2x2 #75	(8)

Ilustración 20: Listado de tareas del sprint 9

2.10. Sprint 10: 28/12/2016 – 3/1/2017



Illustration 21: burndown chart del sprint 10

Se trata de un sprint centrado principalmente en tareas de documentación. En él se desarrollan la mayoría de los capítulos que forman parte de los anexos del proyecto.

Debido a que no existía una plantilla para OpenOffice o LibreOffice orientada a los anexos, como sí ocurría con la memoria, se ha desarrollado una nueva a partir de la plantilla de la que se disponía. Los apartados que componen la plantilla se han seleccionado en base a la plantilla de Latex que sí que estaba disponible.

También se han realizado algunas correcciones en los capítulos 2 y 3.

Finalmente se ha creado una máquina virtual Ubuntu en la que poder integrar el proyecto.

En la ilustración 22 podemos ver el listado de tareas de este sprint.

Completed Issues and Pull Requests	Story points
⌚ Anexos: Plan de proyecto Documentación PlataformaTorneos2x2 #76	(8)
⌚ Anexos: Especificación de requisitos Documentación PlataformaTorneos2x2 #77	(5)
⌚ Anexos: Especificación de diseño Documentación PlataformaTorneos2x2 #78	(5)
⌚ Anexos: Documentación técnica de programación Documentación PlataformaTorneos2x2 #79	(8)
⌚ Anexos: Documentación de usuario Documentación PlataformaTorneos2x2 #80	(8)
⌚ Máquina virtual PlataformaTorneos2x2 #81	(3)
⌚ Crear plantilla para los anexos Documentación PlataformaTorneos2x2 #82	(1)
⌚ Correcciones de los capítulos 2 y 3 Documentación PlataformaTorneos2x2 #84	(1)

Illustration 22: Listado de tareas del sprint 10

2.11. Sprint 11



Illustration 23: burndown chart del sprint 11

Este sprint está centrado en terminar los aspectos pendientes de la documentación y preparar la entrega final del proyecto.

Como podemos ver en el listado de tareas 24, también se ha dedicado parte del sprint a la solución de algunos bugs. Además se han realizado varias correcciones en la documentación.

Completed Issues and Pull Requests	Story points
☒ Conclusiones y líneas de trabajo futuras. Documentación PlataformaTorneos2x2 #70	(5)
☒ Configuración de la máquina virtual PlataformaTorneos2x2 #83	(8)
☒ Documentación del sprint 10 Documentación PlataformaTorneos2x2 #85	(1)
☒ Licencias Documentación PlataformaTorneos2x2 #86	(3)
☒ Redistribución de directorios PlataformaTorneos2x2 #87	(3)
☒ Página de contacto Código PlataformaTorneos2x2 #88	(1)
☒ Añadir al tribunal al repositorio de GitHub PlataformaTorneos2x2 #89	(1)
☒ Bug al migrar a Ubuntu bug Código PlataformaTorneos2x2 #92	(5)
☒ Bug en el login de alumno bug Código PlataformaTorneos2x2 #93	(1)
☒ Tests Documentación Test PlataformaTorneos2x2 #95	(8)
☒ Feedback tras el registro bug Código PlataformaTorneos2x2 #96	(2)
☒ Alerts de nuevo torneo en mozilla bug Código PlataformaTorneos2x2 #97	(1)
☒ Pasar bibliografía al final. Documentación PlataformaTorneos2x2 #98	(1)
☒ Análisis de las licencias del software utilizado. Documentación PlataformaTorneos2x2 #99	(1)
☒ La plantilla de los anexos no recoge bien las imágenes. bug Documentación PlataformaTorneos2x2 #100	(1)
☒ Preparación de la máquina virtual para incluirla en CD PlataformaTorneos2x2 #101	(3)
☒ La llamada a la finalización del torneo no funciona en Ubuntu. bug Código PlataformaTorneos2x2 #102	(1)
☒ Fallos gráficos en Ubuntu bug PlataformaTorneos2x2 #103	(1)

Illustration 24: Listado de tareas del sprint 11

3. Estudio de viabilidad

3.1. Viabilidad legal

A lo largo del proyecto se ha utilizado software externo con licencias propias que tienen influencia sobre el apartado legal del proyecto. Pasemos a analizar cada uno de los casos.

- Bootstrap: está registrado bajo la licencia MIT. Esta licencia permite “usar, copiar, modificar, fusionar, publicar, distribuir, cambiar la licencia y/o vender el software” [1] siempre que se incluya el aviso de la licencia.
 - Bootstrap confirmation [2]: una herramienta que ofrece confirmaciones al usuario

más sencillas a través de Bootstrap. Su licencia es Apache 2.0 [3] y permite utilizarlo, modificarlo, distribuir versiones modificadas, etc, siempre y cuando se incluya la licencia y un aviso de responsabilidad.

- NetLogo: se encuentra bajo licencia GNU GPL [4], que permite usar, copiar, modificar el software y redistribuirlo.

Dado que NetLogo, la herramienta base de este proyecto, está distribuido bajo una licencia GNU GPL, se ha decidido utilizar también esta licencia para este caso. La licencia se adapta bien a las necesidades del proyecto [5], y la versión utilizada es la 3.0 [4].

3.2. Viabilidad práctica

Dado que se trata de una herramienta destinada a la docencia, en lugar de un análisis económico exhaustivo, se va a analizar la viabilidad de la puesta en práctica en un aula de esta herramienta.

En primer lugar analizaremos los costes. Esta es una herramienta destinada a su utilización en materias de nivel universitario en grados relacionados con el campo de la teoría de juegos. Asumimos que en una clase de este tipo tanto los alumnos como el profesor disponen de equipos informáticos. Cualquier computador hoy en día, por simple que sea, puede utilizar un navegador. Por ello, los costes de los usuarios no representan un punto importante en el presupuesto. En el caso del servidor, se da un caso diferente. Se necesita un lugar en el que mantener la aplicación y un dominio para la misma. El precio de un host con php, mysql y java puede costar 10\$ mensualmente.

En cuanto a la puesta en marcha de la herramienta en una clase, se trata de algo bastante asequible. El único requisito para la participación de los alumnos es un nivel mínimo de programación en NetLogo, alcanzable en pocas horas para quien no lo conoce. Dado el tipo de estudios al que va destinado la herramienta, esto no debería suponer un impedimento. Los conocimientos en el lado del creador de los torneos tampoco suponen un gran problema para el profesor, se trata de tareas muy simples que no requieren de ningún conocimiento que no se tenga ya.

Podemos concluir que llevar a las aulas esta herramienta es una tarea muy simple debido a que se trata de una tecnología fácil de utilizar ejecutada sobre una plataforma web, lo que aumenta su disponibilidad.

II - ESPECIFICACIÓN DE REQUISITOS

1. *Introducción*

El objetivo de este apartado es la definición de las características que se desea que la aplicación tenga al final de su desarrollo. Para ello fijaremos los objetivos que se intentan cumplir y, más formalmente, se definirán los requisitos funcionales y no funcionales de la aplicación. Finalmente se profundizará en los casos de uso derivados de la funcionalidad que se desea.

2. *Objetivos generales*

Tal y como se trató en la memoria, el proyecto cuenta con objetivos divididos en las categorías de funcionales, técnicos y académicos. En este caso nos centraremos en los objetivos técnicos y funcionales.

- El principal objetivo de la aplicación es el desarrollo de una herramienta web en la que poder celebrar torneos de tipo 2x2. Se busca que los alumnos desarrollen estrategias y las enfrenten entre ellas para, bajo ciertos parámetros de payoff, obtener una serie de puntuaciones según su éxito en el entorno del torneo. Se busca que estos torneos puedan celebrarse simultáneamente en varias salas independientes.
- Un objetivo importante del desarrollo es que los alumnos dispongan de una herramienta con la que poder probar sus estrategias contra otras ya preestablecidas.
- Se busca establecer un sistema de usuarios en el que se diferencien dos tipos: profesor y alumno. Gracias a este sistema se espera poder almacenar los datos los resultados de los torneos en los que participa cada alumno y establecer un sistema de diferenciación de funcionalidades. Este sistema hará que los profesores puedan realizar ciertas tareas que los alumnos no pueden realizar en la aplicación.
- Una prioridad en los objetivos es permitir cierta variedad en la realización de los torneos. Para ello se desea proporcionar al profesor la posibilidad de establecer diferentes parámetros de payoff en cada torneo celebrado, con el objetivo de que cada uno sea diferente.
- En un ámbito centrado en el uso de la aplicación se busca una interacción del usuario intuitiva y fácil de llevar a cabo. Para ello se tienen como objetivos un diseño responsive, una disponibilidad alta en cuanto a navegadores, una respuesta rápida en la finalización del torneo, evitar la instalación de programas externos, etc.

3. *Catálogo de requisitos*

3.1. *Requisitos funcionales*

La aplicación desarrollada deberá cumplir una serie de requisitos funcionales establecidos en el listado siguiente:

- RF – 1: se debe permitir a los usuarios registrarse en la aplicación, ya sea como alumno (participante) o como profesor (creador de torneos).
- RF – 2: un usuario debe poder identificarse con su email y su contraseña al entrar a la

aplicación.

- RF – 3: la aplicación debe permitir a los usuarios registrados como profesores la creación de torneos.
 - RF – 3.1: el creador del torneo deberá poder fijar los parámetros de payoff deseados en el momento de inicio del mismo.
- RF – 4: se deberá permitir a los usuarios registrados como profesores finalizar los torneos. Solo los usuarios que hayan creado cierto torneo podrán finalizarlo.
 - RF – 4.1: se deberá permitir la ejecución de un torneo global que enfrente a las estrategias que han enviado los alumnos entre ellas.
- RF – 5: todos los usuarios deberán poder participar en un torneo añadiendo una estrategia. Los usuarios que no estén registrados podrán participar añadiendo sus datos, pero perderán la posibilidad de ver torneos anteriores en su perfil.
 - RF – 5.1: la aplicación permitirá a los participantes probar su estrategia contra una serie de soluciones ya establecidas para comprobar su normal funcionamiento antes de enviarla definitivamente.
- RF – 6: los usuarios deberán poder consultar los resultados de los torneos.
 - RF – 6.1: cada participante, si está registrado, podrá ver en su perfil los torneos en los que ha participado anteriormente.

3.2. Requisitos no funcionales

Además de los requisitos funcionales establecidos, la aplicación deberá cumplir los siguientes requisitos no funcionales:

- RNF – 1: el diseño de la aplicación deberá adaptarse a pantallas de diferentes tamaños sin comprometer su funcionalidad.
- RNF – 2: el tiempo de resolución del torneo, a pesar de contar con una carga pesada de trabajo, deberá ser razonable.
- RNF – 3: la aplicación deberá mostrar diferentes opciones en función del tipo de usuario que se haya identificado en ese momento.
- RNF – 4: se deberán poder alojar varias salas con un torneo cada una, con el objetivo de poder celebrar varios torneos al mismo tiempo.

4. Especificación de requisitos

4.1. Diagrama de casos de uso

DIAGRAMA DE CASOS DE USO

Plataforma para la realización de torneos 2x2 | Jose Antonio Barbero

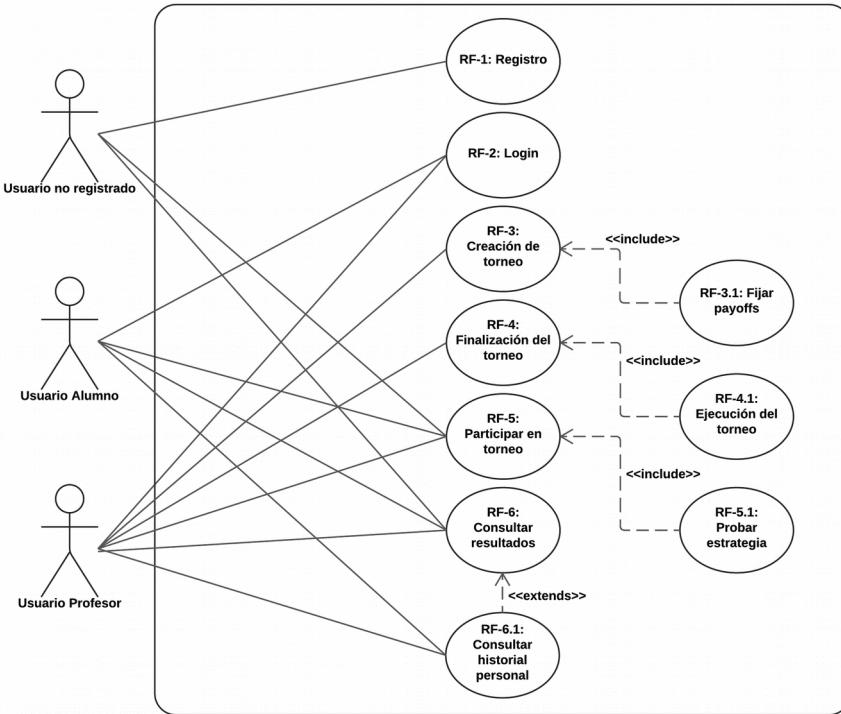


Illustration 25: Diagrama de casos de uso

4.2. Casos de uso

4.2.A. CU1 – Registro

Caso de uso	Registro
Versión	1.0
Autor	Jose Antonio Barbero
Requisitos	RF – 1: se debe permitir a los usuarios registrarse en la aplicación, ya sea como alumno (participante) o como profesor (creador de torneos).
Descripción	El usuario introduce una serie de datos para pasar a formar parte de los usuarios registrados en base de datos.
Precondiciones	<ul style="list-style-type: none"> - El usuario no debe estar registrado. - El email y la contraseña deberán ser válidos. - La contraseña y su confirmación deberán ser iguales.
Acciones	El usuario introducirá sus datos personales, entre los que se encontrarán el email y la contraseña, que serán guardados en base de datos. Finalmente pulsará el botón para enviar el formulario.
Postcondiciones	La contraseña deberá ser cifrada.
Excepciones	En caso de que no se cumplan las precondiciones, no se enviará el formulario y se pedirá que se introduzcan correctamente.
Importancia	Media.

Table 1: Caso de uso 1: Registro

4.2.B. CU2 – Log in

Caso de uso	Login
Versión	1.0
Autor	Jose Antonio Barbero
Requisitos	RF – 2: un usuario debe poder identificarse con su email y su contraseña al entrar a la aplicación.
Descripción	El usuario introduce su email y contraseña para acceder a la aplicación.
Precondiciones	El email y la contraseña deben existir en la base de datos.
Acciones	Introducción del email, introducción de la contraseña y envío del formulario a través del botón.
Postcondiciones	La contraseña deberá ser cifrada.
Excepciones	Si no existe tal usuario, se le notificará para que lo reintente.
Importancia	Media.

Table 2: Caso de uso 2: Login**4.2.C. CU3 - Creación del torneo**

Caso de uso	Creación del torneo
Versión	1.0
Autor	Jose Antonio Barbero
Requisitos	RF – 3: la aplicación debe permitir a los usuarios registrados como profesores la creación de torneos. RF – 3.1: el creador del torneo deberá poder fijar los parámetros de payoff deseados en el momento de inicio del mismo.
Descripción	El profesor será capaz de crear torneos en los que se puedan inscribir los participantes.
Precondiciones	El usuario deberá estar registrado como profesor.
Acciones	El creador del torneo introducirá un nombre para el mismo y unos parámetros de payoff.
Postcondiciones	Ninguna.
Excepciones	Ninguna.
Importancia	Alta.

Table 3: Caso de uso 3: Creación de un torneo

4.2.D. CU4 - Finalización del torneo

Caso de uso	Finalización del torneo.
Versión	1.0
Autor	Jose Antonio Barbero
Requisitos	<p>RF – 4: se deberá permitir a los usuarios registrados como profesores finalizar los torneos. Solo los usuarios que hayan creado cierto torneo podrán finalizarlo.</p> <p>RF – 4.1: se deberá permitir la ejecución de un torneo global que enfrente a las estrategias que han enviado los alumnos entre ellas.</p>
Descripción	El creador de cierto torneo podrá ponerle fin, ejecutando así el modelo de Netlogo asociado y enfrentando a todas las estrategias registradas.
Precondiciones	El usuario que quiere finalizar el torneo deberá ser el mismo que lo creó.
Acciones	El usuario finalizará el torneo haciendo clic en el botón de finalización y aceptando una confirmación.
Postcondiciones	Ninguna.
Excepciones	Si el usuario no es el creador de ese torneo, se le notificará que sólo el creador puede finalizarlo.
Importancia	Alta.

Table 4: Caso de uso 4: Finalizar torneo

4.2.E. CU5 - Participar en el torneo

Caso de uso	Participar en torneo.
Versión	1.0
Autor	Jose Antonio Barbero
Requisitos	<p>RF – 5: todos los usuarios deberán poder participar en un torneo añadiendo una estrategia. Los usuarios que no estén registrados podrán participar añadiendo sus datos, pero perderán la posibilidad de ver torneos anteriores en su perfil.</p> <p>RF – 5.1: la aplicación permitirá a los participantes probar su estrategia contra una serie de soluciones ya establecidas para comprobar su normal funcionamiento antes de enviarla definitivamente.</p>
Descripción	Los usuarios tendrán la posibilidad de unirse a un torneo creando su propia estrategia y probándola contra alguna preestablecidas.
Precondiciones	<ul style="list-style-type: none"> - El email introducido por el usuario debe ser válido. - La estrategia introducida debe compilar en el modelo NetLogo web.
Acciones	<ul style="list-style-type: none"> - El usuario introducirá un nombre para su estrategia y sus datos en caso de que no esté registrado. - A continuación abrirá la parte de código fuente del modelo NetLogo y modificará el código para incluir su estrategia personalizada. - Tendrá la oportunidad de ejecutar el modelo para probar el rendimiento contra varias de las estrategias seleccionables en el modelo.
Postcondiciones	Ninguna.
Excepciones	<ul style="list-style-type: none"> - En caso de que el email sea erróneo, se le notificará al usuario hasta que introduzca uno válido. - En caso de que el modelo del usuario no compile, se le notificará a través de avisos por pantalla indicando qué ha fallado.
Importancia	Alta.

Table 5: Caso de uso 5: Participar en torneo

4.2.F. CU6 - Consultar resultados

Caso de uso	Consultar resultados.
Versión	1.0
Autor	Jose Antonio Barbero
Requisitos	RF – 6: los usuarios deberán poder consultar los resultados de los torneos. RF – 6.1: cada participante, si está registrado, podrá ver en su perfil los torneos en los que ha participado anteriormente.
Descripción	Se ofrecerá la posibilidad de que los usuarios consulten los resultados de los torneos celebrados.
Precondiciones	Para ver los torneos de uno mismo, se debe estar registrado.
Acciones	El usuario accederá al apartado de torneos finalizados para ver todos los torneos o a su perfil para ver los torneos en los que ha participado.
Postcondiciones	Ninguna.
Excepciones	Ninguna.
Importancia	Media.

Table 6: Caso de uso 6: Consultar resultados

III - ESPECIFICACIÓN DE DISEÑO

1. Introducción

Este apartado está destinado al análisis del tipo de distribución de cada una de las partes de la aplicación en el sistema. Comenzaremos estudiando la forma de distribución de los datos para pasar a tratar el flujo de una ejecución de la aplicación, finalizando por el análisis del diseño estructural de la misma.

2. Diseño de datos

Se han compaginado dos formas diferentes de almacenamiento de los datos en este proyecto. La primera, en base de datos, es permanente; mientras que la segunda, en ficheros, es información temporal.

Pasemos a analizar la estructura de la base de datos viendo la información que almacena cada una de las tablas en particular. Los nombres de cada campo dejan bastante claro el tipo de información que contienen así que profundizaremos solamente en aquellos con alguna característica relevante.

- **Users:** es la tabla en la que se almacena la información de los usuarios registrados en la aplicación. Cabe destacar que campo password almacena la contraseña del usuario cifrada y que el campo profesor es un booleano para identificar los privilegios de ese usuario.

v  torneos2x2db users	
id :	int(11)
username :	varchar(50)
nombre :	varchar(50)
apellidos :	varchar(50)
email :	varchar(100)
password :	varchar(100)
#profesor :	tinyint(1)

Illustration 26: Tabla “users” de base de datos

- **Participantes:** almacena la información de los participantes de un torneo concreto, por ello cada uno se identifica con el id del torneo y su número de participante en el mismo. Inicialmente el campo puntuación se inicializa a cero, para actualizarlo cuando se complete el torneo.

v  torneos2x2db participantes	
✉	nombre : varchar(50)
✉	apellidos : varchar(50)
✉	nombreEstrategia : varchar(50)
🔑	idTorneo : int(11)
🔑	numeroParticipante : int(11)
✉	correo : varchar(50)
#	puntuacion : int(11)

Illustration 27: Tabla “participantes” de base de datos

- **Torneos:** almacena la información de cada uno de los torneos creados en la aplicación. El campo idUserCreador se utiliza para que solo quien lo ha creado pueda finalizarlo. El campo finalizado es un booleano utilizado para controlar la forma de mostrar las puntuaciones.

v  torneos2x2db Torneos	
🔑	id : int(11)
✉	nombre : varchar(50)
#	idUserCreador : int(11)
#	finalizado : tinyint(1)
📅	fechalinicio : date
📅	fechaFin : date

Illustration 28: Tabla "torneos" de base de datos

- **Salas:** se trata de una tabla que almacena qué torneo está en cada sala en cierto momento, si es que hay alguno. El valor de idTorneo va variando según se finalizan y crean torneos, y puede ser null si no hay ningún torneo en ese momento.

torneos2x2db Salas	
idSala :	int(11)
#idTorneo :	int(11)

Illustration 29: Tabla "salas" de base de datos

- **Payoffs:** almacena los cuatro valores de recompensa que el tutor quiera fijar en el momento de creación del torneo.

torneos2x2db payoffs	
idTorneo :	int(11)
#mutualcoop :	int(11)
#suckerspayoff :	int(11)
#mutualdefect :	int(11)
#temptationtodeflect :	int(11)

Illustration 30: Tabla "payoffs" de base de datos

El caso de la información temporal almacenada en ficheros responde a la necesidad de importar cierta información al modelo NetLogo. NetLogo, por sí mismo, no soporta la utilización de SQL por lo que se ha decidido mantener esta información en una serie de ficheros que se analizan más en profundidad en el apartado correspondiente al diseño arquitectónico.

3. Diseño procedimental

Dado que un diagrama de flujo de la aplicación completa sería confuso, y que el aspecto más importante es el proceso de desarrollo de un torneo, vamos a analizarlo desde su creación hasta su finalización.

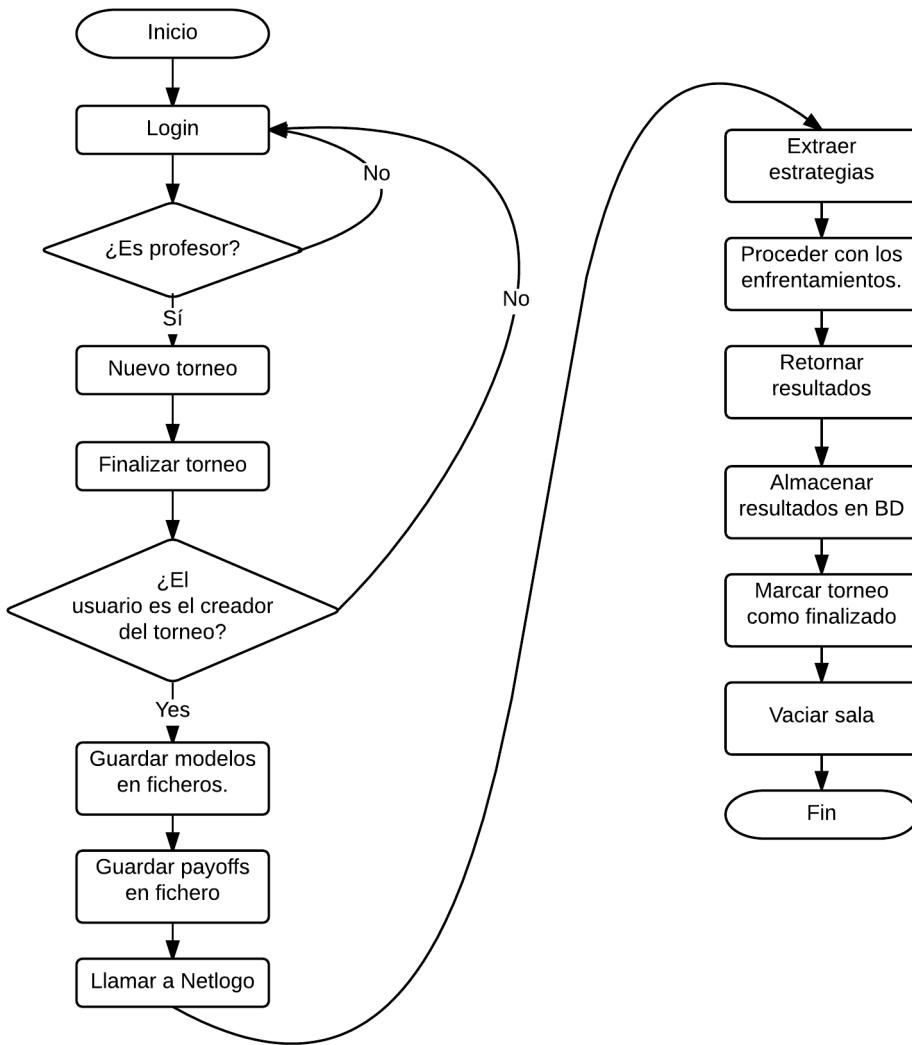


Illustration 31: Diagrama de flujo de la vida de un torneo

4. Diseño arquitectónico

La aplicación está distribuida en una serie de ficheros que se encuentran en los directorios que serán analizados a continuación.

En primer lugar tenemos el directorio base. En él encontramos el fichero index.html necesario para la ejecución de la aplicación web.

Dentro del directorio antes mencionado, encontramos los directorios que analizaremos a continuación. En primer lugar comentaremos tres de ellos: css, js y fonts. Son los tres directorios utilizados por Bootstrap para ofrecer cambios en la apariencia de la aplicación, y algunas funcionalidades en el caso de js. Se han añadido, aparte de los que incluye Bootstrap, algunos estilos propios necesarios dentro del archivo styles. Dentro del apartado javascript se encuentra la carpeta con las funcionalidades java necesarias para NetLogo.

En segundo lugar, encontramos el directorio NetLogo, en él se halla toda la tecnología necesaria para el correcto funcionamiento de los modelos. Encontramos en él, aparte de los propios modelos, individual y global, un directorio “alumnos” con la información que se guarda en ficheros para su ejecución. En él podemos ver una serie de directorios “sala” en los que se almacenan

los modelos de los participantes de cada una. También vemos un directorio en el que se almacenan los modelos de la sala que va a ser ejecutada, y un último directorio en el que se extraen las estrategias de los modelos por parte de NetLogo.

También encontramos un directorio “test” en el que se han almacenado las pruebas realizadas sobre la aplicación.

Finalmente, en el directorio src encontramos todos los fichero html y una carpeta php con el código necesario para el lado del servidor.

4.1. Modelo-vista-controlador

Cabe destacar que se ha seguido en la distribución de la aplicación del patrón modelo-vista-controlador en la distribución de la aplicación.

“Modelo-vista-controlador es un patrón utilizado para aislar la lógica de negocio de la interfaz de usuario. En el uso de modelo-vista-controlador, el modelo representa la información (los datos) de la aplicación y las reglas de negocio utilizadas para manipular los datos; la vista corresponde a los elementos de la interfaz de usuario, como el texto, elementos checkbox, etc. El controlador maneja las acciones del usuario como las acciones del teclado o del ratón, y las conduce hacia el modelo o la vista, según se requiera.” [1]

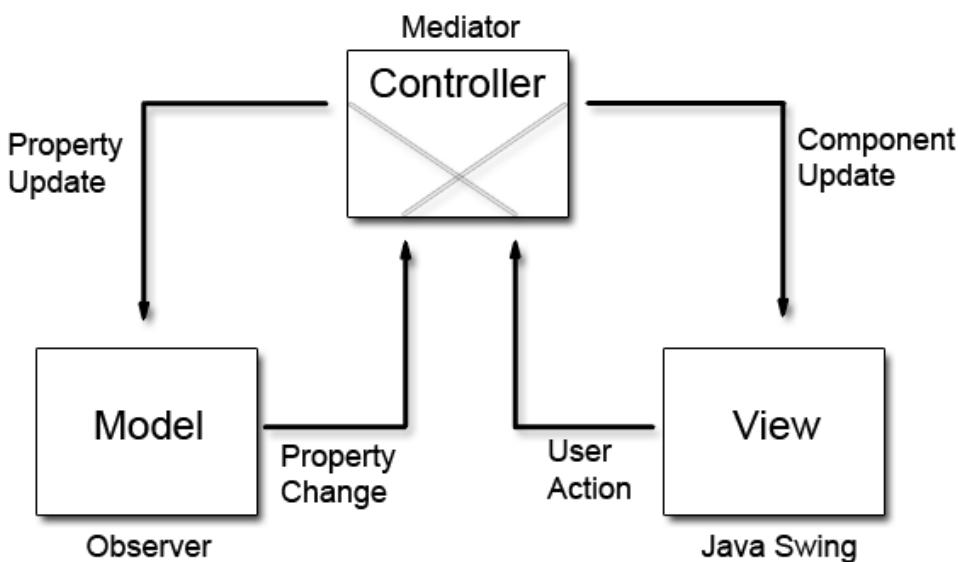


Illustration 32: Modelo-vista-controlador[1]

Como podemos observar, a lo largo de la estructura del proyecto existe una diferenciación de los tres elementos que componen este patrón, veamos dónde se aplica:

- **Modelo:** podemos encontrar los componentes del modelo en el apartado dedicado a NetLogo. Son los elementos que contienen realmente las funcionalidades que buscamos en el proyecto. Igualmente forma parte del modelo la base de datos, aunque no la encontraremos en forma de directorios.

- Vista: la vista se encuentra, tanto en los archivos html como en los componentes de bootstrap css, js y font. Son los elementos que permiten mostrar al usuario la aplicación e interactuar con ella.
- Controlador: los componentes controladores los encontramos en los archivos php, así como en los jar que controlan NetLogo y el ejecutable shell que los lanza. Estos elementos son los encargados de manejar la ejecución y la funcionalidad de la aplicación.

IV - DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

1. *Introducción*

Este capítulo está destinado a explicar aquellos detalles necesarios para comprender el desarrollo del proyecto por parte de un programador externo.

2. *Estructura de directorios*

PlataformaTorneos2x2 es el directorio general del proyecto. En él encontramos las carpetas que analizaremos a continuación y el archivo index.html, que sirve de pantalla inicial para la aplicación. También encontramos el ejecutable que provoca el lanzamiento de NetLogo Behaviour Space.

- BaseDeDatos: incluye un archivo SQL para importar la base de datos del proyecto en caso de que se deseé.
- css: contiene los archivos de estilo en cascada necesarios para Bootstrap, además del archivo styles.css que añade estilos propios y para NetLogo.
- doc: almacena la documentación de la memoria. Se incluyó este directorio para ir viendo el progreso en GitHub, ya que la herramienta de gestión de tareas ZenHub depende de esta plataforma.
- fonts: incluye fuentes incluidas por parte de Bootstrap.
- js: contiene los archivos JavaScript de Bootstrap.
 - netlogojars: contiene la tecnología Java que requiere NetLogo.
- Netlogo: almacena toda la tecnología necesaria para NetLogo. En ese directorio concreto encontramos los modelos global e individual.
 - Alumnos: contiene aquellos elementos que el modelo global necesita importar o exportar.
 - Estrategias_Alumnos: las estrategias extraídas de los modelos son almacenadas en este directorio.
 - Modelos_Alumnos: contiene los modelos de la sala en la que se va a finalizar el torneo.
 - Salas: contienen los modelos, separados por salas, que se obtienen de las respuestas de los alumnos en el modelo web.
- src: contiene los archivos html que conforman las páginas de la aplicación web.
 - php: contiene el código necesario para la programación en el lado del servidor.
- test: en este directorio se han almacenado las pruebas que se han realizado sobre la aplicación.
 - selenium: contiene las pruebas exportadas en java y html

3. Manual del programador

Tomaremos este apartado como una guía ante desarrolladores que pudiesen continuar con el progreso de este proyecto.

Lo primero que necesitaremos será establecer el entorno de desarrollo que se ha seguido en este caso. Varias de las herramientas son sustituibles por otras según las preferencias del desarrollador en cuestión. En especial el editor de código, que se deja a libre elección, aunque en este caso se utilizó SublimeText con el plugin Emmet.

En primer lugar, es necesario tener instalado Java en su versión 8. Para llevar a cabo su instalación en Ubuntu, ejecutaremos los comandos siguientes:

```
sudo apt-get update
```

```
sudo apt-get install default-jre
```

3.1.A. NetLogo

La versión de NetLogo utilizada en el desarrollo del proyecto es la 5.3.1 por ser la más actual en este momento. Para instalarlo en Ubuntu deberemos descargar la versión que deseemos de <https://ccl.northwestern.edu/netlogo/5.3.1/> y descomprimirla donde queramos almacenar NetLogo.

El directorio que genera NetLogo tras su instalación contiene una serie de librerías y archivos “.jar”. Estos son los archivos que necesitaremos para la ejecución en el servidor, para el resto de ejecuciones de los modelos, utilizaremos la versión de escritorio común.

Quizás el aspecto más relevante a destacar sobre la tecnología NetLogo utilizada es la herramienta NetLogo Behaviour Space. Se trata de una utilidad que nos proporciona una forma de realizar experimentos con NetLogo con tantas ejecuciones, parámetros y resultados como necesitemos. Para crear un nuevo experimento NetLogo deberemos acceder a Herramientas>Analizador de comportamiento. Ahí se nos dará la posibilidad de crear un nuevo experimento.

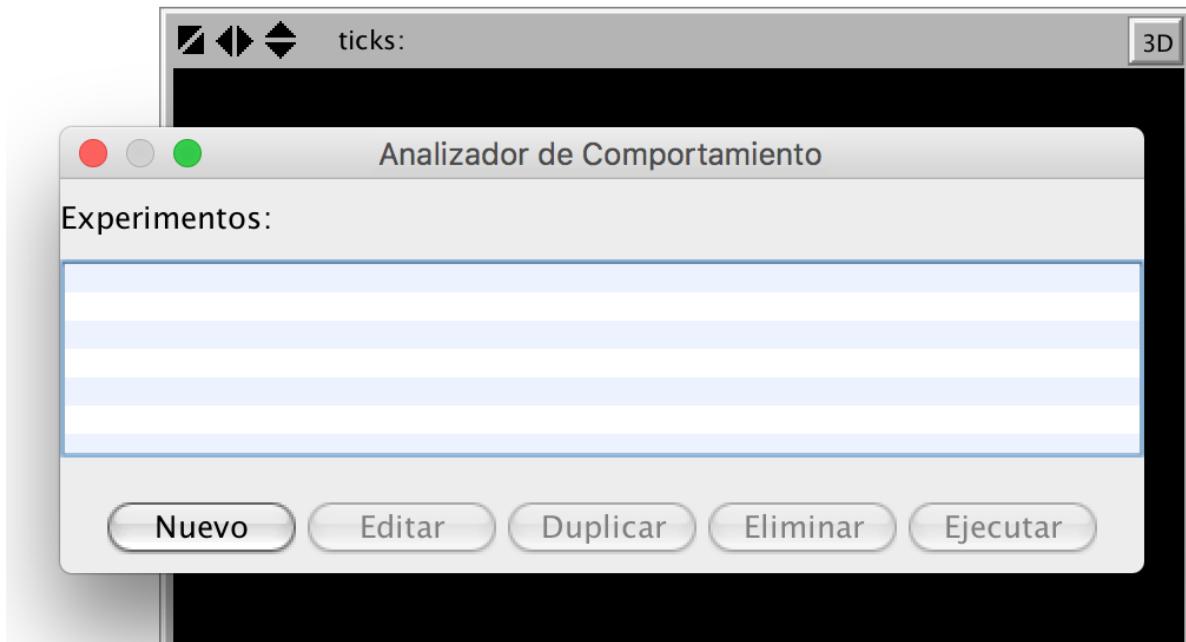


Illustration 33: Nuevo experimento en NetLogo Behaviour Space

Una vez veamos esta pantalla, haremos clic en “Nuevo” y encontraremos el formulario

siguiente:

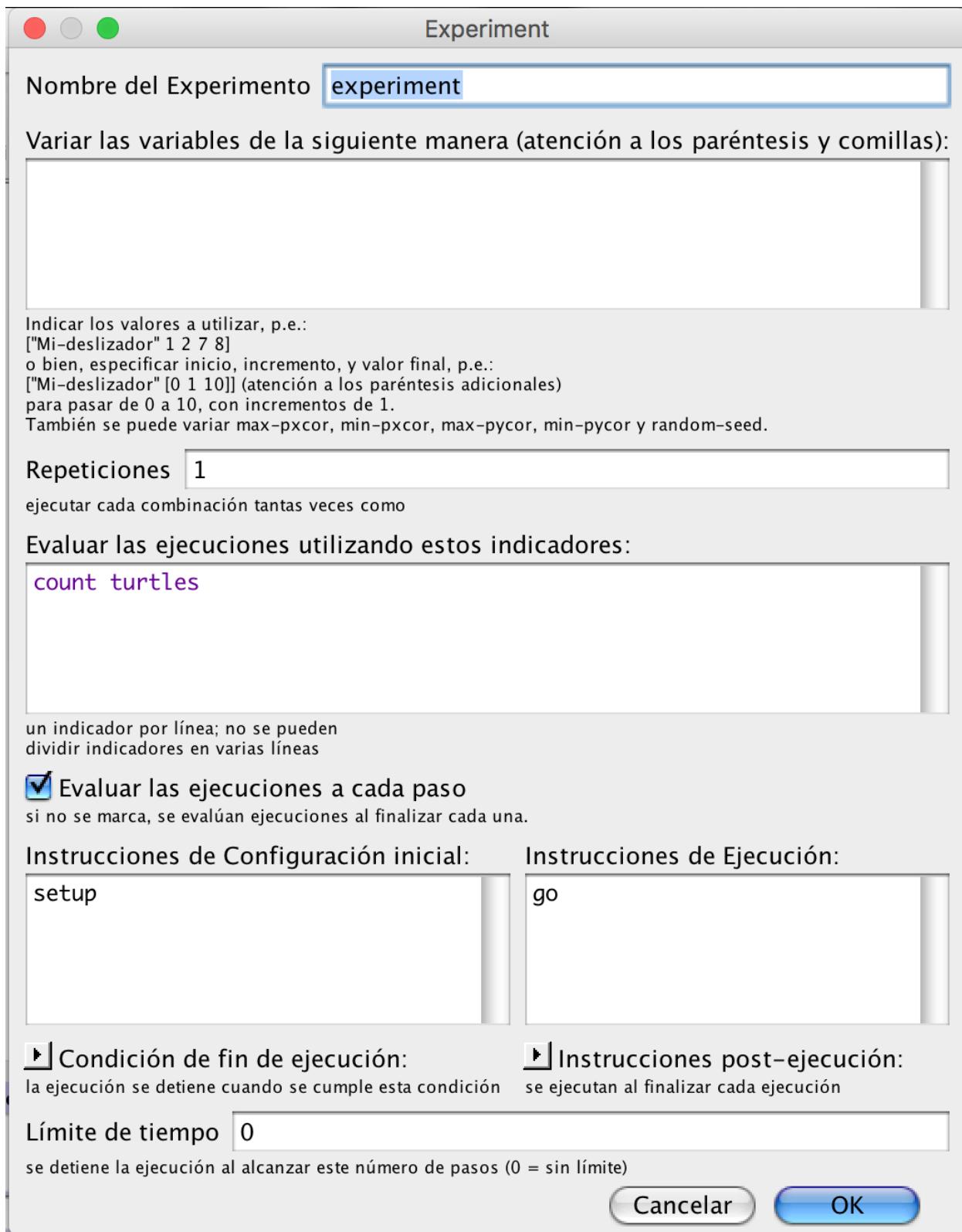


Illustration 34: Formulario de un nuevo experimento en NetLogo Behaviour Space

En él podemos fijar las condiciones que deseemos para el experimento: nombre, forma de modificar las variables, número de veces que queremos que el experimento se repita, forma de evaluar el resultado de una ejecución, método que establece las condiciones iniciales de cada ejecución, método principal del modelo, condiciones iniciales, finales, etc.

Una vez finalizado el formulario del experimento veremos una pantalla igual que la inicial, solo que ahora incluirá nuestro nuevo experimento.

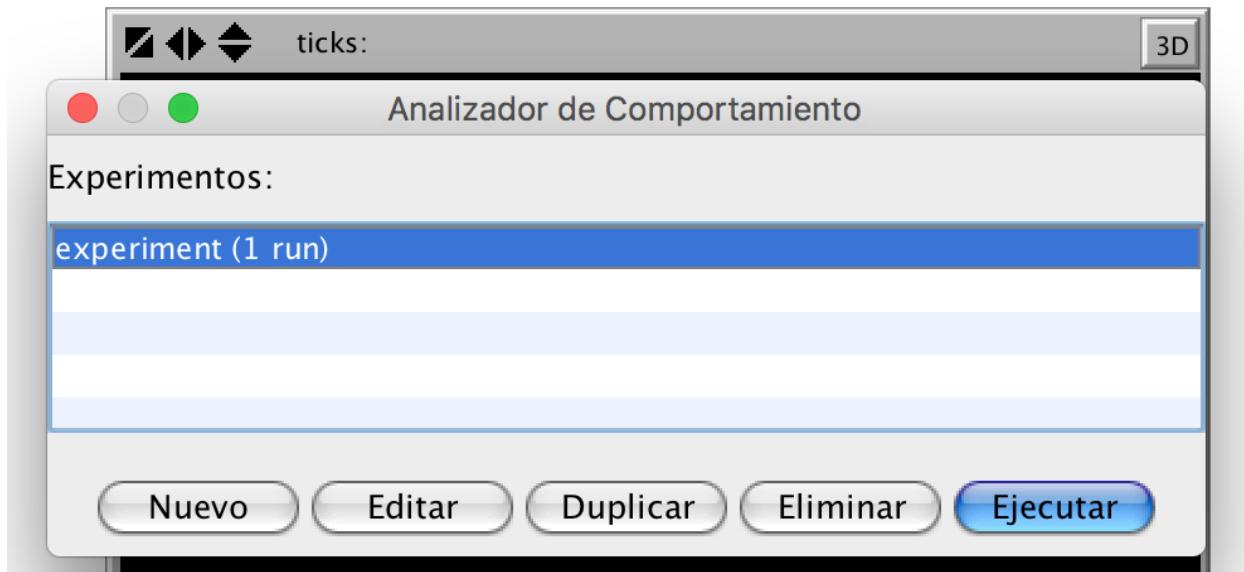


Illustration 35: Experimento creado en NetLogo Behaviour Space

Una vez disponemos de nuestro experimento, podemos ejecutarlo fijando la salida de resultados que deseemos.

En el caso de este proyecto, al ser una ejecución desde el servidor, se ha ejecutado el experimento desde un archivo shell en modo “headless”, es decir, sin apartado gráfico. Si se observa el propio archivo shell se puede entender el sencillo comando que ejecuta.

No es recomendable ejecutar el modelo global independientemente de la aplicación web para evitar cambios de permisos en los archivos que pueden causar problemas posteriores.

3.1.B. XAMPP

Para la creación del servidor local en nuestra máquina se ha utilizado XAMPP. Su instalación, como en el resto de programas utilizados, no supone un problema real. El único aspecto a tener en cuenta es la carpeta `htdocs`, creada en el directorio de instalación, ya que contendrá los archivos de nuestra aplicación. Una vez instalado, deberemos incluir los contenidos del repositorio de código de este proyecto, dentro de la carpeta `htdocs`.

Una vez instalado XAMPP necesitaremos arrancar los servidores, para ello iremos a la pestaña “Manage Servers” y pulsaremos la opción “Start All”, de tal forma que el aspecto de los servidores sea el de la ilustración siguiente.

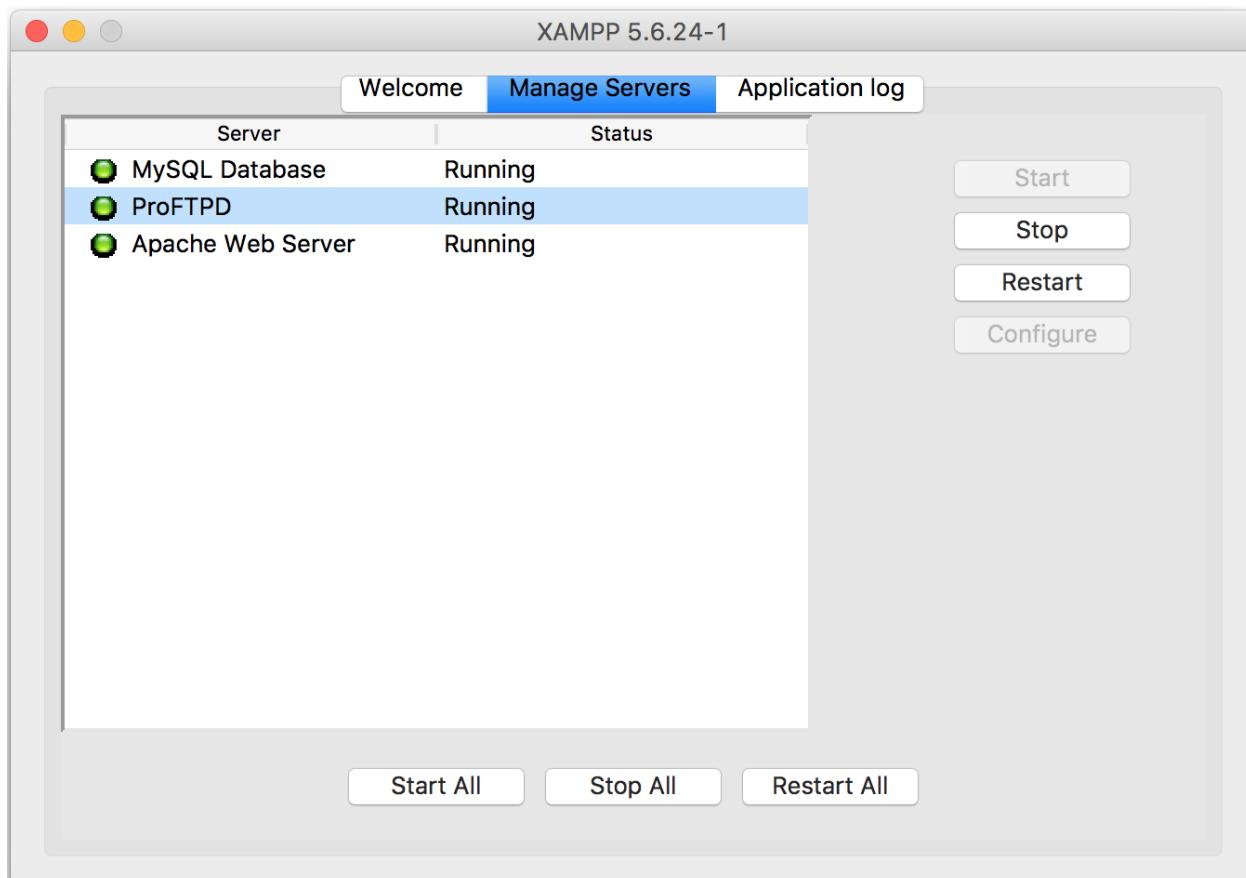


Illustration 36: Servidores iniciados en XAMPP

Para realizar las gestiones necesarias en la base de datos, se ha utilizado phpMyAdmin. Esta herramienta viene incluida con XAMPP y nos permite realizar operaciones SQL (gráficamente y con consultas tradicionales) sobre la base de datos. Un elemento que hay que tener en cuenta es que inicialmente la base de datos estará vacía, para poder contar con la base de datos utilizada en este proyecto se deberá importar el archivo sql que encontramos en la carpeta “BaseDeDatos” incluida en los directorios del proyecto. Para ello, dentro de phpMyAdmin, crearemos una nueva base de datos con el nombre que deseemos. Una vez creada, y con ella seleccionada, iremos a la pestaña importar. En ella se nos pedirá que seleccionemos un archivo, elegiremos el archivo sql mencionado anteriormente, y aceptaremos. Esto creará todas las tablas con los datos necesarios dentro de nuestra nueva base de datos.

Una vez establecido el proyecto, es posible que se encuentren algunos problemas de permisos dependiendo de el sistema en el que se haya creado el proyecto. Para solucionar estos problemas, si es que surgen, se deberá dotar de los permisos necesarios a la carpeta del proyecto con el comando siguiente:

```
sudo chmod -R 777 /opt/lamp/htdocs
```

El comando está orientada a la ruta por defecto en un sistema Ubuntu. Si no es el caso, se puede adaptar el comando a la ruta que se desee y solucionará el problema del mismo modo.

4. Compilación, instalación y ejecución del proyecto

Este apartado está orientado a plantear una ejecución en el entorno de la máquina virtual entregada al tribunal en este trabajo de fin de grado. En caso de que se desee ejecutar la aplicación en otro entorno, se aconseja seguir los pasos del apartado “Manual del programador” para crear un entorno similar.

Dado que se trata de una aplicación web, no es necesaria ninguna instalación ni ninguna compilación, es suficiente con abrir la aplicación desde un navegador, teniendo en cuenta que los servidores de XAMPP estén activos. Para ello, como se ha explicado en el apartado anterior (Illustration 36), debemos ir a la pestaña “Manage Servers” y pulsar en “Start All”.

Una vez iniciado el servidor, solo deberemos entrar en nuestro navegador a la dirección “localhost” y accederemos a nuestra web.

Una vez ejecutado, podremos trabajar con la aplicación, pero eso se explicará más adelante, en el apartado “Manual de usuario”.

5. Pruebas del sistema

Dado que NetLogo no soporta de forma independiente pruebas unitarias, se ha decidido centrar las pruebas del proyecto en la web. Para ello se ha utilizado Selenium IDE. Las pruebas pueden encontrarse en el directorio “test” del proyecto, exportadas en html y java para poder ser importadas a Selenium.

Test Case
Login Bad Email
Login Correcto
Login Bad Password
Registro Correcto
Registro Repetido
Registro Bad User
Registro Bad Name
Registro Bad Apellidos
Registro Bad Mail
Registro Bad Mail2
Registro Bad Password
Registro Bad Confirmation
Registro Bad Match
Inscripción Correcta
Inscripción Bad Name
Inscripción Bad Email
Inscripción Bad Strategy
Nuevo Torneo Correcto
Nuevo Torneo Bad Name
Nuevo Torneo Bad Payoff1
Nuevo Torneo Bad Payoff2
Nuevo Torneo Bad Payoff3
Nuevo Torneo Bad Payoff4

Runs:	23
Failures:	0

Illustration 37: Ejecución de las pruebas.

Encontramos un total de 23 test que ponen a prueba aspectos como el login, el registro o las inscripciones en los torneos, la creación de nuevos torneos. En ellos se ha comprobado la respuesta de la aplicación ante campos vacíos, emails incorrectos, contraseñas incorrectas, contraseñas que no coinciden con su confirmación, valores incorrectos en general, etc. Si se desean con-

sultar los datos que se han utilizado en las pruebas, se pueden observar en los archivos html en forma de tabla, sin necesidad de ejecutar los test completamente.

Se han encontrado algunos errores en el proceso gracias a los test que se han subsanado. Entre ellos, por ejemplo, fallos en las alertas en firefox al crear nuevos torneos o problemas en los mensajes mostrados tras el registro de un usuario.

A parte de las pruebas realizadas con Selenium, en los aspectos en los que resultaba menos óptimo utilizar esta herramienta, se han realizado test de forma manual. Entre estos casos encontramos la finalización de los torneos, en la que había que comprobar manualmente tanto en base de datos como en los ficheros del servidor si se había producido un cambio en el archivo que almacena los resultados, en el estado de los torneos, etc. El mismo caso se da en la participación de un alumno con los archivos guardados en el servidor.

A continuación se pueden observar los casos de prueba utilizados para Selenium y para las pruebas de tipo manual:

Casos de prueba: Login	
Acción	Resultado esperado
Email incorrecto.	"Introduzca un email."
Login correcto.	Perfil visible.
Contraseña incorrecta.	"Usuario o contraseña incorrectos".

Table 7: Casos de prueba en el login

Casos de prueba: Registro	
Acción	Resultado esperado
Registro correcto.	Redirección a los torneos.
Registro repetido.	"Ya existe un usuario con ese email."
Usuario vacío.	Se le pedirá que complete el campo.
Nombre vacío.	Se le pedirá que complete el campo.
Apellidos vacíos.	Se le pedirá que complete el campo.
Email vacío.	Se le pedirá que complete el campo.
Email incorrecto.	Se le pedirá que introduzca un email.
Contraseña vacía.	Se le pedirá que complete el campo.
Confirmación no coincide con contraseña.	"Las contraseñas no coinciden".

Table 8: Casos de prueba en el registro

Casos de prueba: Inscripción.

Acción	Resultado esperado
Inscripción correcta.	Redirección a participación.
Nombre vacío.	Se le pedirá que complete el campo.
Email incorrecto.	Se le pedirá que complete el campo.
Nombre de estrategia vacío.	Se le pedirá que complete el campo.
Inscribirse en una sala llena.	“La sala no admite más participantes”

Table 9: Casos de prueba en la inscripción a los torneos

Casos de prueba: Finalizar Torneo

Acción	Resultado esperado
Finalizar torneo correctamente.	Cambio en base de datos del estado del torneo y de las puntuaciones (también en fichero).
Sala vacía.	“No existe ningún torneo en ese sala”
Finalizar torneo que haya creado otro.	“Debe ser el creador de un torneo para poder finalizarlo.”

Table 10: Casos de prueba en la finalización de torneos

Casos de prueba: Nuevo torneo

Acción	Resultado esperado
Nuevo torneo correcto.	Alerta de torneo creado correctamente.
Nombre vacío.	Se le pedirá que complete el campo.
Payoff vacío.	Se le pedirá que complete el campo.
Payoff no numérico.	Se le pedirá que introduzca un número.

Table 11: Casos de prueba en la creación de torneos

Casos de prueba: Probar Estrategia

Acción	Resultado esperado
Entregar estrategia correctamente.	Cambio en el fichero de esa sala.
Estrategia con fallos de código.	Se mostrará un error en el modelo NetLogo.

Table 12: Casos de prueba en la fase de test de la estrategia

V - DOCUMENTACIÓN DE USUARIO

1. *Introducción*

El objetivo del contenido de este apartado es explicar a un posible usuario todos los aspectos que necesita conocer, desde la instalación hasta las opciones de uso finales de la aplicación.

2. *Requisitos de usuarios*

2.1. Requisitos software

Al tratarse de una aplicación web, no se requiere de componentes software especialmente complejos, dado que el navegador es suficiente para la mayoría de las tareas. En el caso de que deseemos simular el servidor en uno local, necesitaremos software adicional.

- Sistema operativo: el proyecto ha sido desarrollado para MacOS Sierra y es compatible con Ubuntu hasta su versión 16.
- Navegador web: se requiere de un navegador web ordinario. La aplicación está desarrollada sobre Google Chrome, pero puede utilizarse sobre otros, como Mozilla Firefox, si se prefiere.
- Java 8: si la aplicación va a utilizarse con un servidor local, es necesario instalar Java para la ejecución de NetLogo. Se recomienda la versión 8 por ser la utilizada en el desarrollo del proyecto.
- XAMPP: para la creación del servidor local es necesario un servicio que proporcione esta funcionalidad. Se recomienda XAMPP por ser el utilizado a lo largo del desarrollo, pero se puede utilizar cualquier otro.
- No es necesaria la instalación de NetLogo, ya que con los archivos java incluidos en el proyecto es suficiente.

2.2. Requisitos hardware

Por la naturaleza del proyecto, no se requiere una máquina especialmente potente para su ejecución. Por ello, los requisitos hardware estarán marcados por las exigencias de los programas que utiliza[1]–[4].

- Intel Pentium 4 o superior.
- 1Gb de espacio en disco.
- 512 MB de RAM.

Si lo que se desea es importar la máquina virtual que incluye el proyecto, los requisitos son algo diferentes, debido a las características de la máquina virtual creada.

- Intel Pentium 4 o superior.
- 8Gb de espacio en disco.
- 4Gb de RAM.

3. Instalación

En este apartado estudiaremos paso a paso la instalación de las aplicaciones antes mencionadas sobre un sistema Ubuntu.

3.1. XAMPP

El primer paso es descargar la versión que deseemos de la página oficial de descarga de XAMPP[5][6]. Una vez descargado XAMPP deberemos abrir el terminal y ejecutar el comando siguiente para convertir el archivo a ejecutable:

```
sudo chmod +x xampp-linux-x64-5.6.8-0-installer.run
```

Una vez ejecutado el comando anterior, lanzaremos el ejecutable con el comando siguiente:

```
sudo ./xampp-linux-x64-5.6.8-0-installer.run
```

A continuación observaremos el progreso de la instalación de XAMPP, solo deberemos esperar a que termine. Cuando finalice la instalación el programa se abrirá, y deberemos acceder a la pestaña “Manage Servers” para utilizar la opción “Start All” y con ello arrancar el servidor local. También es posible iniciar el servidor desde el terminal con el comando:

```
sudo /opt/lampp/lampp start – Para iniciar
```

```
sudo /opt/lampp/lampp stop – Para finalizar
```

A continuación deberemos importar la base de datos tal y como se ha explicado en el apartado “Manual del programador” anterior.

3.2. Java

La instalación de Java 8 [7] en Ubuntu es muy sencilla y solo requiere de los comandos siguientes:

- `sudo add-apt-repository ppa:webupd8team/java`
- `sudo apt-get update`
- `sudo apt-get install oracle-java8-installer`

3.3. Navegador

Dado que se ha utilizado Chrome [8] para el desarrollo del proyecto, se va a descargar su versión de Ubuntu para la utilización del mismo.

Para ello, deberemos ejecutar los comandos siguientes:

- `wget -q -O - https://dl-ssl.google.com/linux/linux_signing_key.pub | sudo apt-key add -`
- `sudo sh -c 'echo "deb http://dl.google.com/linux/chrome/deb/ stable main" >> /etc/apt/sources.list.d/google-chrome.list'`
- `sudo apt-get update`

- `sudo apt-get install google-chrome-stable`

4. Manual de usuario

Este manual está orientado a un uso desde la máquina virtual que incluye el proyecto. Si se dispone de la máquina virtual no es necesaria ninguna instalación explicada en apartados anteriores, se deberá seguir la guía siguiente.

Lo primero que se deberá hacer será importar la máquina virtual a través de VirtualBox. En la entrega del trabajo se incluye un archivo “.ova”. Se debe pulsar en “Archivo/Importar Servicio Virtualizado” y seleccionar la ruta del archivo mencionado. Una vez seleccionado pulsaremos “siguiente” y después en “importar”, y esperaremos a que la máquina virtual sea creada.

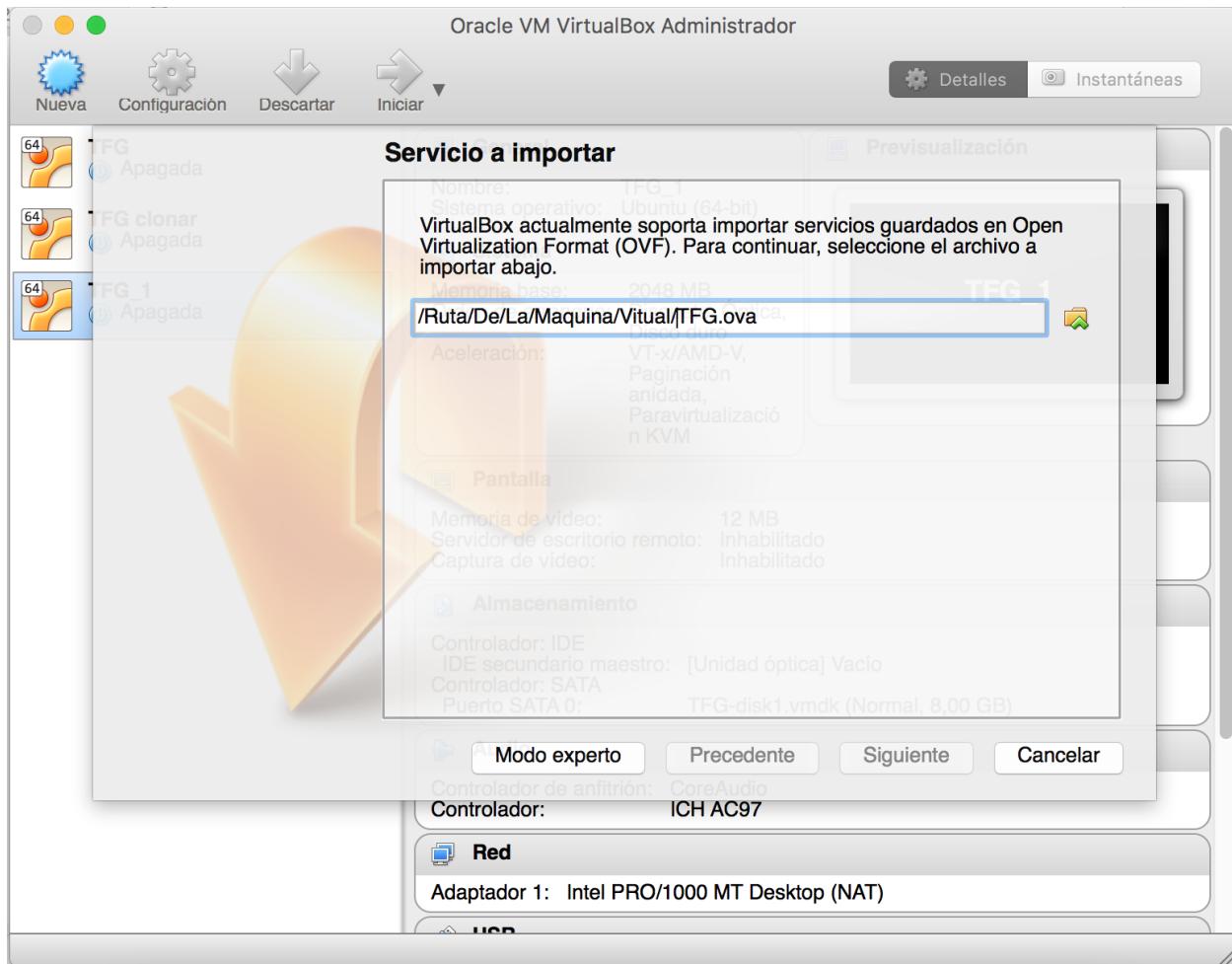


Illustration 38: Importar máquina virtual

Una vez creada la máquina virtual, debemos tener en cuenta las credenciales de usuario siguientes:

Usuario: tfguser Contraseña: tfgpassword

Para poder ejecutar la aplicación en el navegador será necesario levantar el servidor local. Para ello se ejecutará en el terminal el comando siguiente:

```
sudo /opt/lampp/lampp start
```

Si posteriormente se desea parar el servidor, podrá hacerse con el comando:

```
sudo /opt/lampp/lampp stop
```

Dado que ya tenemos el servidor activo, podremos acceder a la aplicación a través del navegador que seleccionemos, a través de la dirección “localhost”.

Con el objetivo de explicar al usuario final todas las funcionalidades de la aplicación, vamos a realizar una serie de ejemplos observando la propia interfaz de la plataforma.

Antes de comenzar, se han creado dos usuarios para permitir probar el funcionamiento de la aplicación. El primero es de tipo alumno y el segundo de tipo profesor, para comprobar ambas funcionalidades.

Usuario de tipo alumno:

Email: alumno@alumno.com - Contraseña: alumno

Usuario de tipo profesor:

Email: profesor@profesor.com - Contraseña: profesor

4.1. Inicio

Lo primero que encontramos al iniciar la aplicación es una pantalla inicial desde la que podemos acceder a la mayoría de las funcionalidades disponibles. La única función que tiene esta página en sí misma es mostrar un resumen del contenido de la aplicación y cuántos participantes están participando en ese momento en cada sala. Pasemos a analizar las funcionalidades a las que podemos acceder desde este nivel, comenzando por la identificación de los usuarios.

The screenshot shows the homepage of the 'Plataforma de Torneos 2x2'. At the top, there's a dark navigation bar with tabs for 'Plataforma de Torneos 2x2' (selected), 'Inicio' (current page), 'Torneos', and 'Inicio de sesión'. Below the header, the main title 'Plataforma de Torneos 2x2' is displayed in large bold letters, followed by the subtitle 'Trabajo de fin de Grado, Jose Antonio Barbero, Universidad de Burgos'. On the right side of the page, there's a sidebar titled 'Torneos' which lists five tournaments: 'Torneo 1' (0/25), 'Torneo 2' (0/25), 'Torneo 3' (0/25), 'Torneo 4' (0/25), and 'Torneo 5' (1/25).

¿Cuál es el propósito de este proyecto?

Las nuevas tecnologías están llevando a la docencia a incorporar ciertas metodologías basadas en herramientas digitales. La utilización de experimentos en forma de juegos en el aula ha demostrado ya buenos resultados. El paso siguiente, los simuladores y juegos ejecutados en computadores, están dando resultados del mismo tipo.

A su vez, la teoría de juegos es un campo que sigue creciendo en la actualidad. Además, la naturaleza de sus conceptos encaja a la perfección con una forma de enseñanza basada en simuladores o juegos.

Del mismo modo, vemos que la enseñanza online está experimentando un crecimiento en expansión gracias, tanto a las universidades con enseñanza a distancia, como a los portales web basados en cursos, como Coursera o Mirida X.

Por estas razones se ha planteado el desarrollo de una herramienta docente de simulación virtual online destinada a la explicación de la teoría de juegos a los alumnos. Se centra en el planteamiento de juegos de tipo 2 x 2, el tipo de juego del conocido dilema del prisionero, un caso ampliamente utilizado para ejemplificar la teoría de juegos. El objetivo es que un profesor pueda crear torneos en los que puedan participar sus alumnos desde sus ordenadores. El profesor tiene libertad para fijar los parámetros del torneo. Tanto los alumnos como el profesor podrán consultar sus puntuaciones al final del torneo para sacar sus conclusiones. El objetivo que se fija a los alumnos es que desarrollen su propia estrategia en forma de código con la que obtener mayor puntuación que el resto de participantes. Se les proporcionará un medio con el que probar esa estrategia contra algunas ya preestablecidas.

Con este proyecto se desea dotar a los profesores y a los alumnos de una manera de afrontar la teoría de juegos desde las aulas más amena e ilustrativa para los estudiantes, y con más oportunidades para los docentes.

Illustration 39: Página de inicio

4.2. Log in

La plataforma dispone de un sistema de sesiones para alumnos y para profesores. Estos podrán hacer clic en la parte superior derecha para iniciar sesión con su correo electrónico y su contraseña, tal y como vemos en la imagen.

Estar identificados otorgará a los usuarios las ventajas de poder consultar en su perfil los torneos pasados y de evitar tener que llenar ciertos datos al participar en un torneo. Sin embargo, un usuario no registrado puede participar en los torneos introduciendo esos datos manualmente.

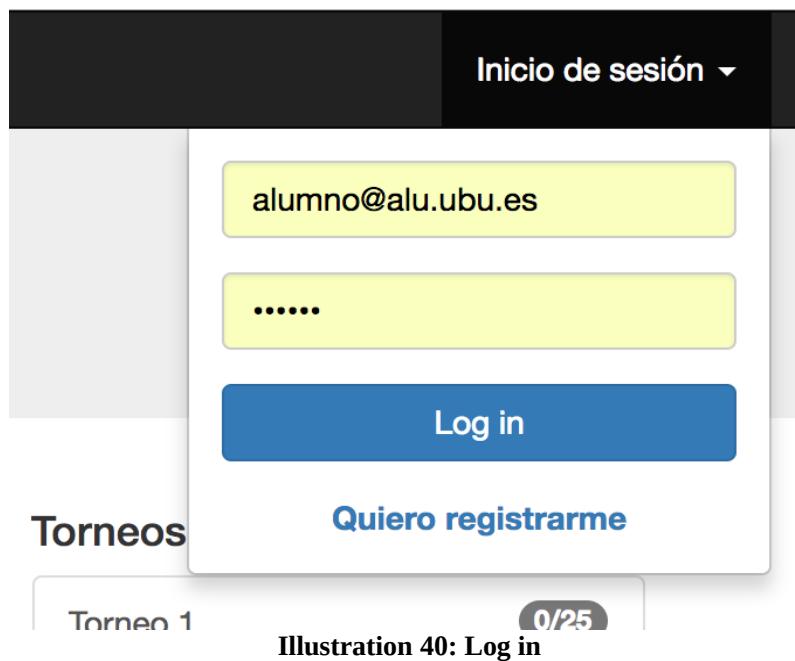


Illustration 40: Log in

4.3. Registro

Como es lógico, antes de poder realizar el paso anterior, si no se tiene una cuenta creada, el usuario debe registrarse utilizando el vínculo “Quiero registrarme” en el panel de log in. Para ello solo deberá llenar un sencillo cuestionario con sus datos y su tipo de cuenta. Podemos observar este cuestionario en la imagen siguiente.

The image shows a registration form titled "Formulario de Registro". The form consists of several input fields: "Nombre de Usuario" (with placeholder "Nombre de usuario"), "Nombre" (with placeholder "Nombre"), "Apellidos" (with placeholder "Apellidos"), "Email" (with placeholder "Email"), "Contraseña" (with placeholder "Contraseña"), and "Confirmar contraseña" (with placeholder "Escriba de nuevo su contraseña"). Below these fields is a "Rol" section with two radio buttons: "Alumno" (which is checked) and "Profesor". At the bottom is a blue rectangular "Enviar" button.

Illustration 41: Registro

4.4. Torneos

También podemos acceder a la pantalla de torneos, en la que podremos observar el formulario a llenar para participar en un torneo. Además, podemos ver en pantalla los torneos que hay en cada sala en ese momento, con el número de participantes que tiene cada uno. Finalmente,

vemos una lista de torneos finalizados desde la que podremos acceder al historial de estos.

Existen dos vistas diferentes de esta pantalla, los usuarios registrados como profesores tienen visibles las opciones de finalización de torneo y la de crear un torneo nuevo, como vemos en la segunda ilustración. Mientras tanto, los usuarios registrados como alumnos no verán estas opciones. La diferencia entre alumnos registrados y los no registrados es que el formulario aparecerá ya llenado para los primeros y vacío para los segundos.

Torneo	Estado	Opciones
Torneo 1	0/25	Finalizar Torneo
Torneo 2	0/25	Finalizar Torneo
Torneo 3	0/25	Finalizar Torneo
Torneo 4	0/25	Finalizar Torneo
Torneo 5	1/25	Finalizar Torneo

Torneos Finalizados
Torneo de prueba - 0000-00-00
nombreTorneo - 0000-00-00
nombreTorneo - 0000-00-00
- 0000-00-00

Illustration 42: Torneos desde la perspectiva de alguien sin registrar

4.5. Crear torneo

Solamente los usuarios registrados como profesores podrán acceder a esta funcionalidad. Se trata de un mecanismo muy simple en el que el profesor rellena un formulario que determina el tipo de torneo que se va a crear a través de los parámetros de payoff.

4.6. Torneos finalizados

Como se ha comentado anteriormente, si los usuarios hacen click en los torneos finalizados de la pantalla “torneos”, serán dirigidos a una página de historial en la que podrán consultar los torneos que se hayan celebrado anteriormente.

Además, los usuarios registrados disponen de una página similar para sus propios torneos a través de su perfil de usuario, en la parte superior derecha.

4.7. Participar

Es una de las principales funcionalidades de la aplicación. Para que un participante se una a un torneo, debe llenar el formulario en el apartado “torneos”, como ya se ha explicado con anterioridad. Una vez se supere ese formulario se entrará en la pantalla que vemos en la imagen siguiente.

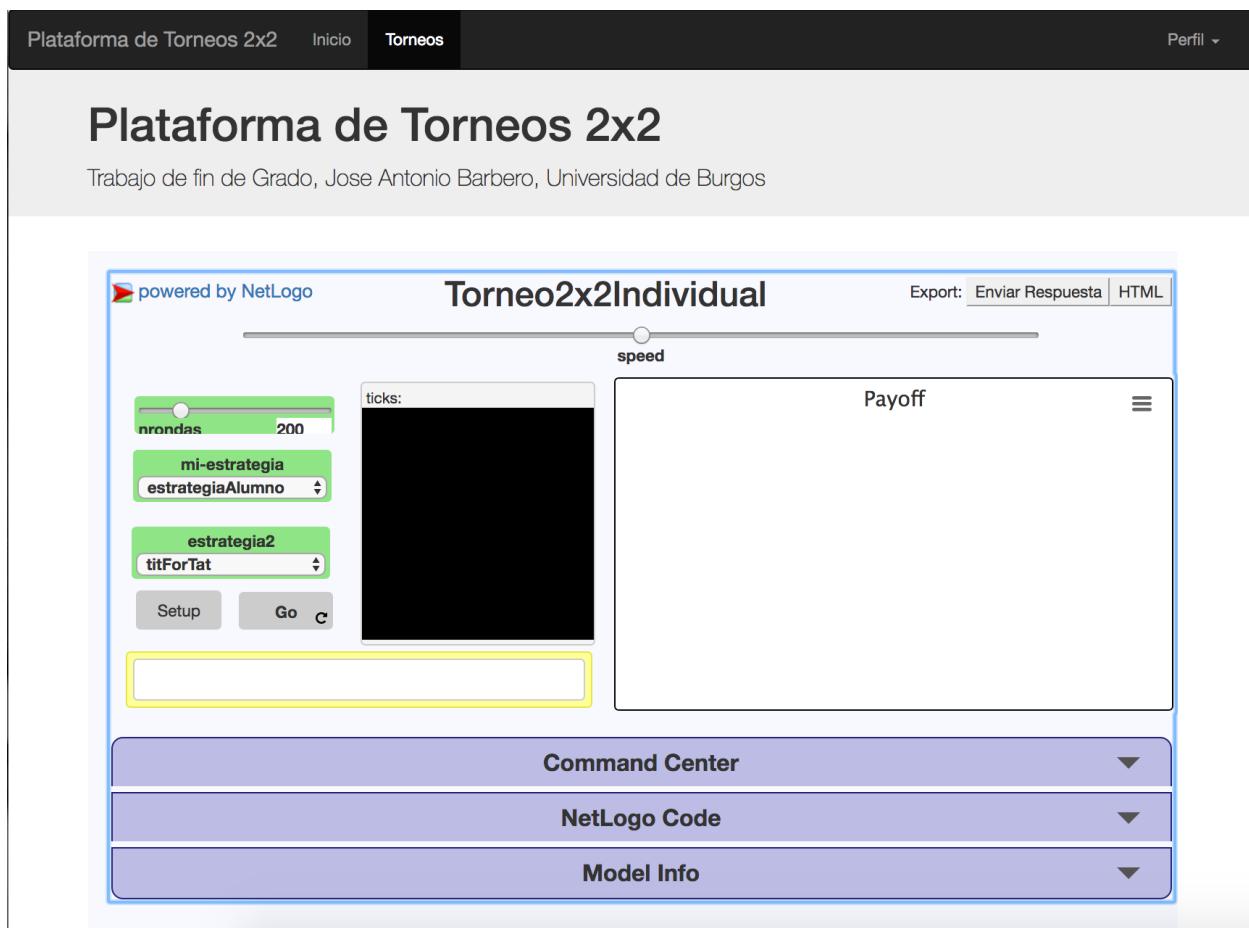


Illustration 44: Participación en el torneo

Se trata del modelo NetLogo sobre el cual los participantes desarrollarán su estrategia. En él los estudiantes pueden probar su estrategia contra varias estrategias que están ya establecidas. Para ello pueden elegirlas entre las opciones de los selectores de fondo verde.

Para la ejecución del modelo se deberá pulsar primero el botón setup y posteriormente el botón go. Una vez hayan finalizado las rondas estipuladas, veremos una gráfica y una salida de la puntuación en forma de texto.

Pero para ello el alumno debe desarrollar una estrategia. Para ello deberá pulsar en “NetLogo Code. Encontrará un código comentado, con un fragmento delimitado en el que podrá introducir su estrategia, como vemos en la imagen siguiente. Las condiciones están explicadas en un

comentario en la parte superior del método.

Para comprobar que el código es correcto antes de la ejecución el usuario deberá compilar el código a través del botón “Recompile code”.

The screenshot shows the Command Center interface. At the top, a grey bar says "Command Center". Below it, a blue bar says "NetLogo Code". In the center, there's a button labeled "Recompile Code". The main area contains the following NetLogo code:

```

19  cooperate_vs_defect
20  defect_vs_cooperate
21  defect_vs_defect
22 ]
23
24 to-report estrategiaAlumno
25   let eleccion False
26   ; PARTICIPANTE, ESCRIBE AQUÍ EL CÓDIGO DE TU ESTRATEGIA
27   ; !!! - No se permite modificar el código fuera de las líneas marcadas.
28   ; 1 - Puedes acceder a las decisiones propias o del oponente a través del campo past-decisions.
29   ; 2 - El método debe retornar la decisión que tomará la estrategia en la ronda
30   ;      actual a través de la variable elección. Sea True si colabora o False si no lo hace.
31   ;-----| INICIO |-----
32
33
34
35
36
37
38   ;-----| FIN |-----
39   report eleccion
40 end
41
42 to setup
43   clear-all
44
45   * Payoffs

```

Below the code, another blue bar says "Model Info".

Illustration 45: Código de la estrategia de un alumno

Una vez se haya probado la estrategia, el alumno deberá enviarla. Para ello pulsará el botón “Enviar Respuesta” en la parte superior derecha del modelo. Se ofrece la opción de descargar el modelo con la estrategia del alumno, por si se quiere utilizar aparte de la plataforma web. También se puede exportar el modelo como Html desde el botón “HTML”.

Finalmente, independientemente de que se descargue el modelo o no, el alumno habrá enviado una respuesta que quedará almacenada para participar en ese torneo.

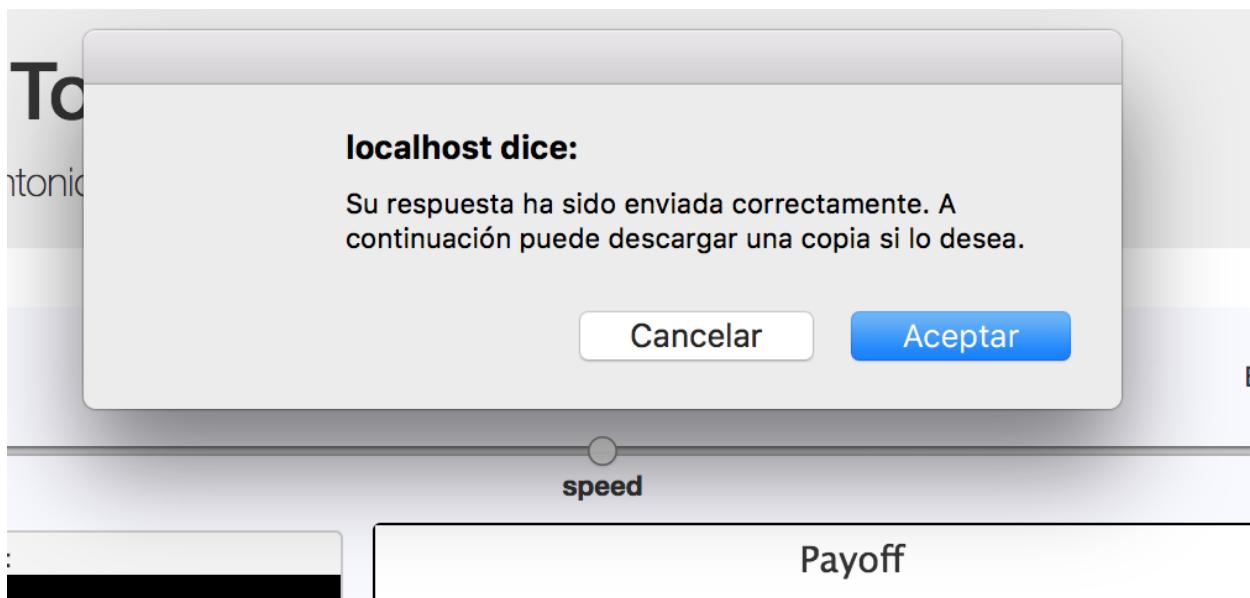


Illustration 46: Descarga del modelo

4.8. Finalizar torneo

Una vez más se trata de una funcionalidad exclusiva de los usuarios registrados como profesores, y les permite finalizar el torneo de cierta sala. Solamente el profesor que haya creado un torneo podrá finalizarlo. Con el objetivo de evitar confusiones, se ha añadido una confirmación en forma de pequeño diálogo para que el profesor se asegure de que desea finalizarlo realmente.

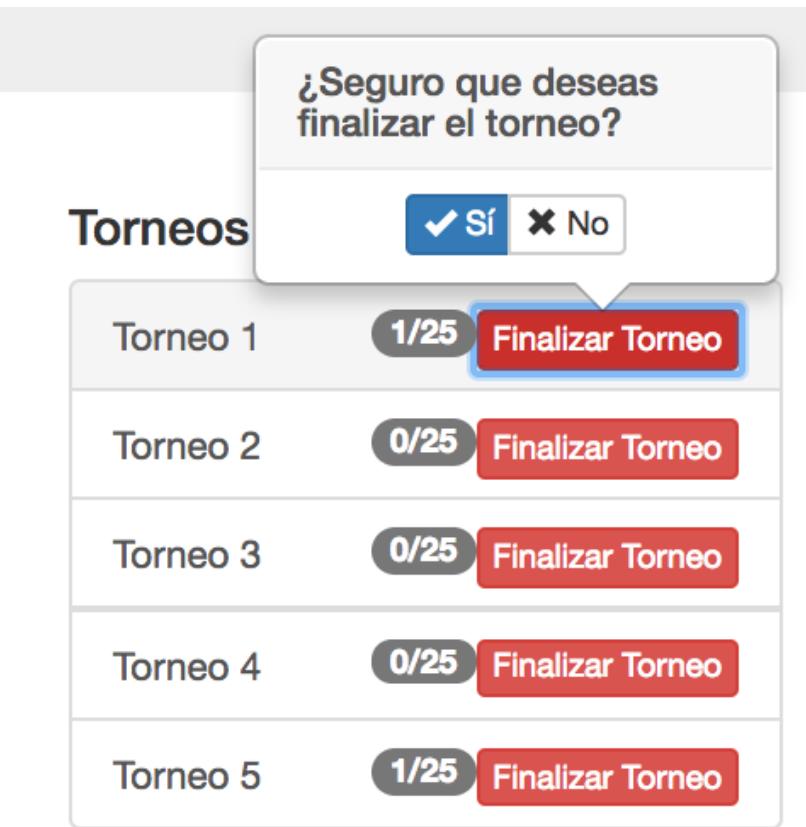


Illustration 47: Finalización de un torneo

VI - REFERENCIAS

Bibliografía

Anexo 1:

- [1] “The MIT License | Open Source Initiative.” [Online]. Available: <https://opensource.org/licenses/MIT>. [Accessed: 09-Jan-2017].
- [2] “Bootstrap Confirmation.” [Online]. Available: <http://bootstrap-confirmation.js.org/>. [Accessed: 09-Jan-2017].
- [3] “Apache License, Version 2.0.” [Online]. Available: <http://www.apache.org/licenses/LICENSE-2.0>. [Accessed: 09-Jan-2017].
- [4] “GNU GPL 3.0.” [Online]. Available: <https://www.gnu.org/licenses/gpl-3.0.txt>. [Accessed: 08-Jan-2017].
- [5] J. G. P. and B. Edmonds, “Open Access for Social Simulation,” 2007.

Anexo 3:

- [1] “Simple Example of MVC (Model View Controller) Design Pattern for Abstraction - CodeProject.” [Online]. Available: <https://www.codeproject.com/articles/25057/simple-example-of-mvc-model-view-controller-design>. [Accessed: 01-Jan-2017].

Anexo 6:

- [1] “XAMPP - Browse /XAMPP Windows/1.8.0 at SourceForge.net.” [Online]. Available: [https://sourceforge.net/projects/xampp/files/XAMPP Windows/1.8.0/](https://sourceforge.net/projects/xampp/files/XAMPP%20Windows/1.8.0/). [Accessed: 03-Jan-2017].
- [2] “Chrome system requirements - Chrome for business and education Help.” [Online]. Available: <https://support.google.com/chrome/a/answer/7100626?hl=en>. [Accessed: 03-Jan-2017].
- [3] “¿Cuáles son los requisitos del sistema para Java?” [Online]. Available: <https://www.java.com/es/download/help/sysreq.xml>. [Accessed: 03-Jan-2017].
- [4] “NetLogo 6.0 User Manual: System Requirements.” [Online]. Available: <https://ccl.northwestern.edu/netlogo/docs/requirements.html>. [Accessed: 03-Jan-2017].
- [5] “How To Install XAMPP Stack On Ubuntu 15.04 | Unixmen.” [Online]. Available: <https://www.unixmen.com/how-to-install-xampp-stack-on-ubuntu-15-04/>. [Accessed: 03-Jan-2017].
- [6] “XAMPP Installers and Downloads for Apache Friends.” [Online]. Available: <https://www.apachefriends.org/es/index.html>. [Accessed: 15-Dec-2016].
- [7] “Cómo instalar Oracle Java 8 en Ubuntu 16.04 – @lobo_tuerto.” [Online]. Available: <http://lobotuerto.com/blog/2014/08/26/como-instalar-oracle-java-en-ubuntu/>. [Accessed: 03-Jan-2017].
- [8] “Install Google Chrome in Ubuntu 14.04 / 13.10 / 12.04 PPA.” [Online]. Available: <http://www.howopensource.com/2011/10/install-google-chrome-in-ubuntu-11-10-11-04-10-10-10-04/>. [Accessed: 03-Jan-2017].

